VŠB TECHNICKÁ UNIVERZITA OSTRAVA
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Detekce anomálií

Anomaly detection

Bc. Bára Groňová

Diplomová práce

Vedoucí práce: Ing. Jan Gaura, Ph.D.

Ostrava, 2021

Abstrakt

Tématem této diplomové práce je detekce anomálií ve videosekvencích pomocí strojového učení. Primárně se zaměřuje na analýzu stavu řidiče vozidla. V úvodu je nastíněna problematika a uplatnění takového systému. Následující teoretická část se zabývá jednotlivými fázemi detektoru anomálií a možnými přístupy k jejich řešení. Je rozdělena do tří základních částí, kterými jsou detekce objektu zájmu, reprezentace dat a analýza dat. Následuje soupis prací souvisejících s tímto tématem. Praktická část je zaměřená na vlastní implementaci detektoru anomálií řidiče vozidla. Zahrnuje popis použitých dat, volbu postupů v jednotlivých částech, shrnutí navržené metody a dosažené výsledky. V závěru práce je popsáno možné využití aplikace, její nedostatky a možnosti dalšího vývoje.

Klíčová slova

detekce anomálií; hluboké učení; RNN; VAE

Abstract

The topic of this master thesis is detection of anomalies in video sequences using machine learning. It primarily focuses on the analysis of the vehicle driver's condition. The introduction outlines the problematics and application of such a system. Following theoretical part deals with the phases of the anomaly detector and possible approaches to their solution. It is divided into three main parts, which are detection of the object of interest, data representation and data analysis. Following is a list of publications related to this topic. Practical part is focused on own implementation of the vehicle driver anomaly detector. It includes description of used dataset, choice of approach in each part, summary of the proposed method and the results achieved. At the end of the work, the possible use of the application, its shortcomings and possibilities for further development are described.

Keywords

anomaly detection; deep learning; RNN; VAE

Poděkování

Ráda bych na tomto místě poděkovala vedoucímu práce Ing. Janu Gaurovi, Ph.D. za odborné vedení a cenné rady při jejím psaní. Díky patří také mé rodině a blízkým za podporu v průběhu celého studia.

Obsah

Seznam použitých zkratek a symbolů

2D	–	Two Dimensional
3D	–	Three Dimensional
ACC	–	Accuracy
AE	–	Autoencoder
AP	–	Average Precision
API	–	Application Programming Interface
CNN	–	Convolutional Neural Network
DNN	–	Deep Neural Network
ECG	–	Electrocardiography
EEG	–	Electroencephalogram
ERT	–	Ensemble of Regression Trees
FPS	–	Frames per second
GAN	–	Generative Adversarial Network
GPU	–	Graphics Processing Unit
GRU	–	Gated Recurrent Units
HOG	–	Histogram of Oriented Gradients
KL	–	Kullback–Leibler
LBP	–	Local Binary Patterns
LSTM	–	Long Short Term Memory
ms	–	millisecond
MSE	–	Mean Squared Error
NaN	–	Not a Number
PAF	–	Part Affinity field
PCA	–	Principal Component Analysis
px	–	pixel
RCL	–	Recall
RNN	–	Recurrent Neural Network
ROC	–	Receiver Operating Characteristic

RoI	–	Region of Interest
RPN	–	Region Proposal Network
SSD	–	Single Shot Detector
VAE	–	Variational Autoencoder
WHO	–	World Health Organization

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Detekce anomálií v lidském chování má široké využití, především pak při zajištění bezpečnosti. Aplikace, které detekují anomálie z videosekvence v reálném čase můžou upozornit na neobvyklé chování lidí na veřejných místech jako jsou letiště či banky, nebo hlídat stav řidiče vozidla. Tato práce se primárně zaměřuje na poslední zmíněné, tedy na detekci anomálií v chování řidiče. Za anomální můžeme považovat např. dušení se, ztrátu vědomí, infarkt, nebo také používání mobilního telefonu. Obecně tedy jakékoli jiné chování než je běžné a bezpečné při řízení vozidla.

Podle světové zdravotnické organizace (WHO) zemře každý rok při autonehodě v průměru 1,35 milionu lidí [1]. Jedná se tak o osmou nejčastější příčinu smrti. Automobilový průmysl, výzkumné instituce i státní zařízení proto vyvíjejí technologie ke snížení počtu nehod, nebo alespoň zmírnění jejich následků. Zdravotní stav a pozornost řidiče vozidla jsou bezpochyby důležitými aspekty bezpečnosti na cestách. Proto se stále více výzkumů zabývá sledováním aktivity řidiče a detekováním neobvyklých, či nebezpečných stavů. Systémy vozidla by pak mohly řidiče upozornit, např. při detekování usínání, případně přebrat řízení a bezpečně zastavit.

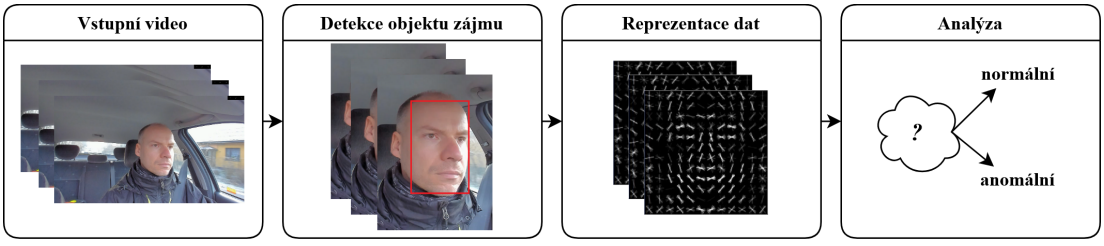
V úvodu práce je nejprve rozebrána problematika detekce anomálií v lidském chování se zaměřením na řidiče vozidla a teoretické základy. Následuje shrnutí prací souvisejících s tímto tématem, jejich výsledky a uplatnění. Další část zahrnuje vlastní implementaci detektoru anomálií řidiče vozidla z videosekvencí a dosažené výsledky. V závěru jsou shrnuty poznatky, možné uplatnění a hlavní nedostatky navrženého detektoru anomálií.

Kapitola 2

Analýza problému

Sledování a analýza lidského chování se dá rozdělit do tří částí. Tou první je detekce člověka z obrazového vstupu, případně jen jeho části, např. obličeje. Následně je potřeba data vhodně reprezentovat a nakonec provést samotnou analýzu. Zjednodušený diagram je na obrázku 2.1. Jednotlivé části na sobě závisí. Pro různé postupy analýzy dat je potřeba zvolit vhodnou reprezentaci a s tím související detekci objektu. Při volbě metod, které budou použité pro jednotlivé části aplikace, je nutné zohlednit mnoho faktorů. Např. pro aplikace, které mají běžet v reálném čase je nutné, aby proces zpracování, analýzy a vyhodnocení byl co nejkratší. Zároveň se musí brát v potaz na jakém zařízení má výsledná aplikace běžet, tedy paměťové a výpočetní limity, a další faktory typické pro danou úlohu.

Při řešení detekce anomálií v chování řidiče vozidla je nutné vzít v potaz výše zmíněnou potřebu aplikace běžet v reálném čase. Pokud by kamera snímala snímek každou desetinu sekundy (FPS 10), celý proces detekce a analýzy by musel být kratší než 0,1 sekundy. Rychlejší metody pro jednotlivé části aplikace často bývají méně spolehlivé (není to však pravidlem). Je tedy nutné určit FPS vstupního videa tak, aby byla kontinuálnost informace dostačující pro analýzu a zároveň bylo možné využít spolehlivých metod. Také je potřeba minimalizovat falešně pozitivní detekce, tedy ty, kdy aplikace chybně označí chování jako anomální. V takovém případě by byl totiž řidič bezdůvodně vyrušován od řízení.



Obrázek 2.1: Zjednodušený diagram procesu detekce anomálií.

Tato kapitola je rozdělena do tří výše zmíněných části. Každá obsahuje různé přístupy k jejich řešení, jejich výhody a nevýhody.

2.1 Detekce člověka

První nezbytnou úlohou je detekce člověka v obraze. Videa z běžného života mají členité pozadí, různé osvětlení či šum. Z toho je potřeba detekovat pouze část, která se má analyzovat, v tomto případě tedy člověk. S ohledem na účel aplikace může být potřeba detekovat všechny lidi v obraze. Při sledování řidiče vozidla je bodem zájmu pouze on a je potřeba jej v obraze správně najít. Aplikace také může používat jen část člověka, nejčastěji obličej. Toho se využívá při sledování výrazu, např. pro detekci usínání. Detekce lidí ve videu je stěžejní v mnoha typech aplikací. Kromě detekce anomálního chování se uplatňuje např. u charakterizace lidské chůze, počítání lidí na veřejných místech, určování pohlaví, či u detekce pádu. Díky tomuto širokému uplatnění se detekcí zabývá mnoho výzkumů a existují tak poměrně spolehlivé metody. Obecně se dají rozdělit na metody založené na extrakci příznaků a metody využívající neuronové sítě.

2.1.1 Metody založené na extrakci příznaků

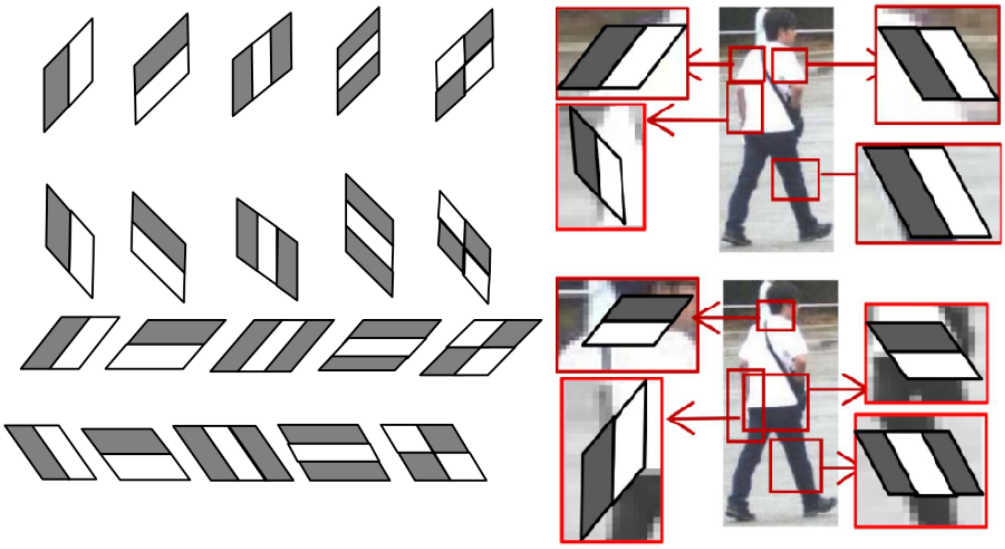
Detekce objektu je obecně rozdělená do dvou základních částí. Nejprve je potřeba z obrazu získat vektor příznaků, který jej popisuje. Následně jsou tyto příznaky klasifikovány, zda obraz obsahuje hledaný objekt, nebo ne. Mezi nejpoužívanější metody pro extrakci vektorů příznaků patří histogram orientovaných gradientů (HOG) a Haarovy příznaky.

- **Haarovy příznaky**

  V roce 1998 Papageorgiou a kol. [2] představili práci se souborem příznaků vycházejícím z Haarových vlnek. Následně v roce 2001 Paul Viola a Michael Jones [3] adaptovali tuto myšlenku a vytvořili framework pro detekci obličeje, který využívá tzv. Haar-like features, neboli Haarovy příznaky. Ty se počítají na konkrétním místě v detekčním okně tak, že se sečtou intenzity pixelů v sousedních obdélníkových oblastech (bíla a černá) a vypočítá se rozdíl mezi nimi. Tento rozdíl se poté použije ke kategorizaci podsekcí obrázku. U problému detekce obličeje jsou oblasti kolem očí světlejší než oblast pod očima a oblast mezi očima je světlejší než oblasti očí. Proto jsou pro detekci obličeje používány Haarovy příznaky tak, jako jsou zobrazeny na obrázku 2.2. Haarovy příznaky mohou být umístěny na kterékoli místo ve sledovaném snímku a můžou mít libovolnou velikost. Ke snížení výpočetní složitosti používá Viola a Jones ve své práci výpočet příznaků za pomoci integrálního obrázku. Pro výběr příznaků použili AdaBoost algoritmus, který iterativním procesem učení tvoří silný klasifikátor jako lineární kombinaci vážených slabých klasifikátorů. Slabým klasifikátorem se zde myslí samostatný obdélníkový příznak.

Obrázek 2.2: Haarovy příznaky používané k detekci obličeje. V prvním řádku jsou zobrazené samotné příznaky a pod nimi jejich umístění na tváři [3].

Poté co Viola a Jones popularizovali využití Haarových příznaků pro detekci obličeje, začaly se v obměnách používat v dalších oblastech. Jejich detektor byl úspěšně rozšířen pro detekci pohybujících se lidí v práci [4]. V oblasti detekce chodců, autoři práce [5] prezentovali detektor lidí ve statických snímcích založený na Haar příznacích, které popisují jednotlivé části člověka (hlava, nohy, ruce), v kombinaci s SVM klasifikátorem. V [6] autoři představili upravený soubor Haarových příznaků, který lepé odráží tvar chodců. Tyto modifikované příznaky jsou zobrazeny na obrázku 2.3.



Obrázek 2.3: Na obrázku vlevo je upravený soubor Haarových příznaků, který lepé odráží tvar chodců. Vpravo je zobrazená aplikace příznaků na snímky chodce. [6]

- **Histogram orientovaných gradientů**

  HOG deskriptor se soustředí na strukturu či tvar objektu. Vizualizace HOG deskriptoru je na obrázku 2.4. Oproti hranovým vektorům příznaků zahrnuje také informaci o směru hrany. Při výpočtu HOG deskriptoru je obrázek nejprve rozdělený do buněk o určité velikosti. Pro každou buňku se vypočítá histogram orientací gradientů. Gradient se počítá aplikací následujících kernelů v horizontálním a vertikálním směru:

$$D_X = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}, D_Y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \tag{2.1}$$

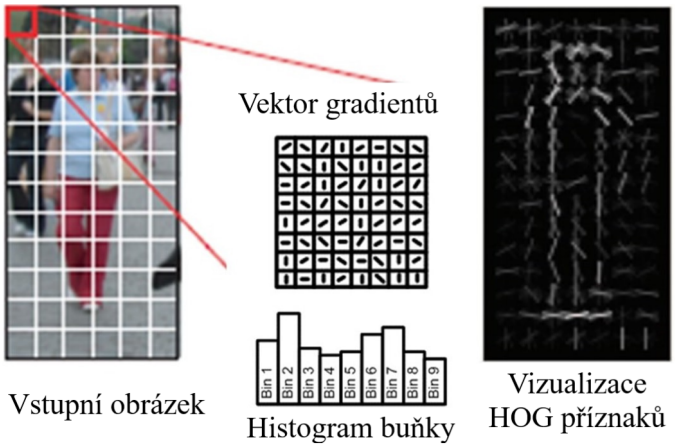
$$I_X = I \star D_X, I_Y = I \star D_Y \tag{2.2}$$

  kde I představuje zpracovávaný obraz. Velikost gradientu (G) a jeho směr (θ) je definován následovně:

$$\mid G \mid = \sqrt{I_X^2 + I_Y^2} \tag{2.3}$$

$$\theta = \arctan \frac{I_Y}{I_X} \tag{2.4}$$

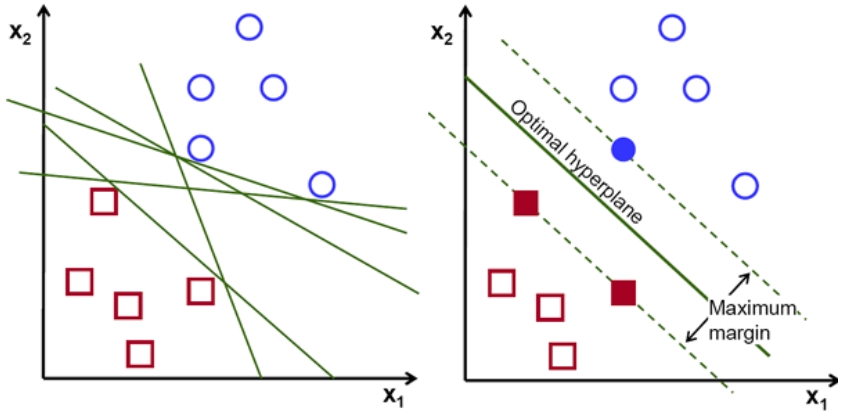
  Histogram je tvořen sloty rozloženými od 0 do 180, nebo do 360 stupňů. Na základě vypočtené orientace je přičtena vážená hodnota do patřičného slotu. Jako váha může být použita velikost gradientu. Histogramy jsou následně normalizované v rámci bloků, které sestávají z určitého počtu sousedních buněk. Vektor příznaků je zřetězení těchto normalizovaných histogramů. Výše zmíněné vzorce ukazují základní postup pro výpočet deskriptoru.



Obrázek 2.4: Vizualizace HOG příznaků [7].

Klasický HOG deskriptor má nevýhodu velkého počtu příznaků což způsobuje, že trénování a klasifikace trvá dlouhou dobu. Další nevýhodou je, že tento deskriptor není invariantní vůči rotaci, což bylo motivací k vytvoření několika variant. Mnoho metod založených na HOG deskriptorech bylo představeno v uplynulých letech. V [8] autoři aplikovali analýzu hlavních komponent (PCA) na HOG vektor k získání tzv. PCA-HOG vektoru, který obsahuje podsoubor původních příznaků. Ten následně použili jako vstup pro SVM klasifikátor. Jejich metoda byla použitá pro detekci chodců a dosáhla uspokojivých výsledků. Felzenszwalb a kol. [9, 10] představili dílčí detektor, kde jsou objekty reprezentovány pomoci několika částečných HOG modelů, které jsou trénovány diskriminační metodou. Metoda dosáhla skvělých výsledků v oblasti detekce a klasifikace objektů.

Na základě získaných vektorů příznaků je potřeba určit, zda se jedná o hledaný objekt. Nejčastějším algoritmem pro tuto úlohu je SVM (Suport Vector Machine), jehož účelem je najít optimální nadrovinu, která rozděluje prostor příznaků na odlišné třídy. Optimální nadrovina je taková, která má co největší prostor od všech bodů obou tříd, jako je znázorněno na obrázku 2.5 [11].



Obrázek 2.5: Některé možné nadroviny (vlevo) a vybraná optimální algoritmem SVM (vpravo) [11].

2.1.2 Metody využívající neuronové sítě

Metody založené na neuronových sítích jsou schopné provádět tzv. end-to-end detekci, což znamená, že jsou schopné provádět kompletní detekci bez specificky definovaných příznaků, jak tomu bylo u předchozích metod. Typicky jsou založené na konvolučních neuronových sítích (CNN). CNN je typ neuronové sítě, která se skládá ze tří typů vrstev: Konvoluční, Pooling a Klasifikační. Konvoluční vrstvy provádí operaci konvoluce nad daty z předchozí vrstvy. Hlavním účelem je získání potřebných vlastností ze vstupu. Konvoluce se provádí pomocí filtrů, které se v průběhu trénování modelu mění a tím se síť učí dosahovat lepších výsledků. Pooling vrstvy mají za úkol zmenšit vstup. To se provádí pro snížení výpočetní náročnosti a pro výběr důležitých příznaků. Za konvoluční vrstvy se zařazují

plně propojené vrstvy, které provádí klasifikaci. Konvoluční sítě je potřeba trénovat na dostatečném množství vhodných dat.

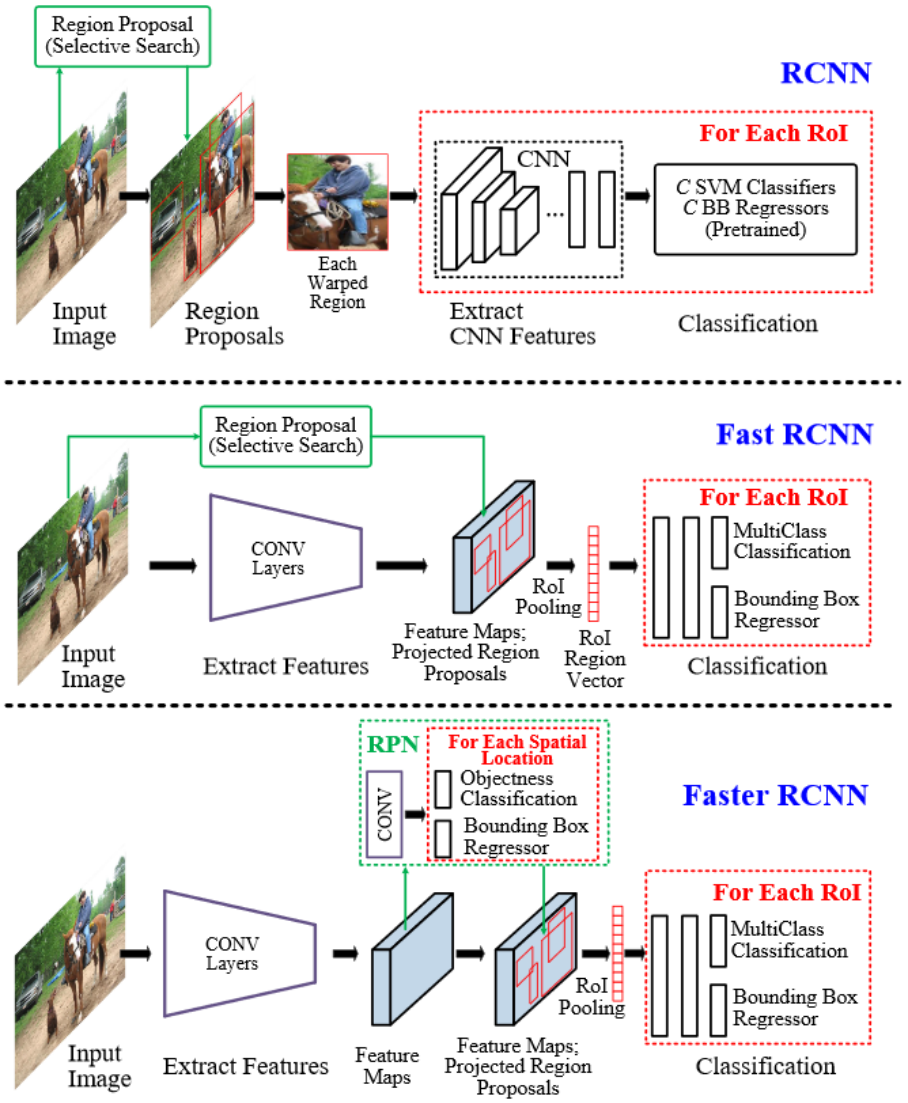
Autoři průzkumu detektorů objektů založených na hlubokém učení z roku 2019 [12] dělí detektory na dvě skupiny. Jsou jimi dvoufázové, které mají vysokou přesnost, a jednofázové, které dosahují vysoké rychlosti. Pro aplikace běžící v reálném čase jsou proto vhodnější detektory jednofázové. Oba typy obsahují páteřní síť, která extrahuje příznaky a dále RoI (Region of Interest) Pooling vrstvu, která řeší problém požadavku na pevnou velikost vektoru pro poslední klasifikační část detektoru. Páteřní síť slouží jako extraktor příznaků ze vstupního obrazu. Za tímto účelem se obvykle používají modely sítí určených ke klasifikaci obrazu, ze kterých jsou odebrány klasifikační plně propojené vrstvy. Můžou být použity hlubší sítě jako např. ResNet [13], AlexNet [14] či VGG [15], nebo menší sítě např. ze třídy sítí MobileNets [16], které jsou vhodné pro mobilní a vestavěné aplikace. Srovnání úspěšnosti vybraných metod popsaných níže v této části nad datovou sadou MS COCO [17] je v tabulce 2.1.

- **Dvoufázové detektory**

  První fáze detektoru navrhuje ohraničující rámečky kandidátů na hledaný objekt. Druhá fáze pak pro každý navrhovaný rámeček extrahuje příznaky pomocí RoI Pooling vrstvy. Ty jsou následně použité pro klasifikaci. Girshick a kol. představili R-CNN detektor založený na regionech [18]. Jejich práce byla první, která ukázala, že použití CNN u detekce objektů nad datovou sadou PASCAL VOC [19] vede ke značnému zvýšení úspěšnosti oproti metodám založeným na extrakci definovaných příznaků. R-CNN detektor obsahuje čtyři moduly. První generuje návrhy regionů nezávisle na kategorii, pro které druhý modul extrahuje vektory příznaků. Následuje soubor SVM klasifikátorů pro jednotlivé třídy. Poslední je regresor pro přesnou predikci ohraničujícího rámečku. O rok později stejný autor navrhl rychlejší metodu nazvanou Fast R-CNN [20]. Oproti původní R-CNN, která provádí průchod CNN sítí pro každý navrhovaný region, Fast R-CNN extrahuje příznaky pro celý vstupní obraz a z nich následně identifikuje regiony. Za použití RoI Pooling vrstvy se změní velikost jednotlivých vektorů příznaků pro navrhované regiony na definovanou velikost a pomocí plně propojených vrstev proběhne klasifikace a lokalizace objektu. Obě výše zmíněné metody využívají pro hledání regionů selektivní vyhledávání, které je pomalé a výrazně ovlivňuje výkon sítě. Shaoqing Ren a kol. představil algoritmus nazvaný Faster R-CNN [21], který místo selektivního vyhledávání využívá pro predikci regionů zájmů separátní RPN (Region Proposal Network) síť. Diagramy výše zmíněných metod jsou znázorněny na obrázku 2.6.

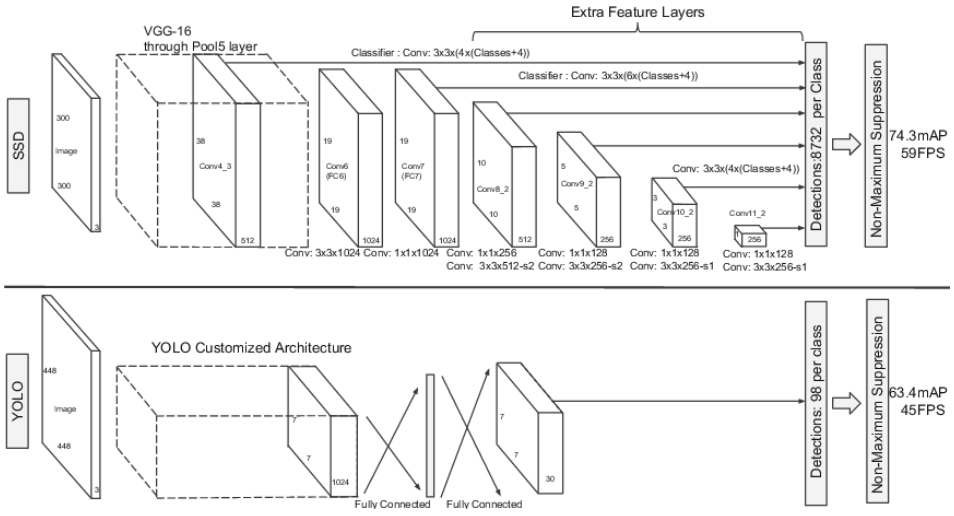
- **Jednofázové detektory**

  Jednofázové detektory predikují rámečky kolem objektů přímo ze vstupního obrazu. Díky tomu jsou rychlejší a vhodné pro použití u aplikací běžících v reálném čase. Jedním zástupcem tohoto typu detektorů je SSD (Single Shot Detector) [23]. Ten jako síť pro extrakci příznaků



Obrázek 2.6: Diagramy algoritmů R-CNN, Fast R-CNN a Faster R-CNN pro detekci objektu v obraze [22].

ze vstupního obrazu používá VGG-16 bez klasifikační vrstvy. Následují konvoluční vrstvy s postupně se snižující velikostí. Z každé z nich jde vektor příznaků do klasifikátoru, což pomáhá detekovat objekty o různé velikosti. Architektura SSD detektoru je v horní části obrázku 2.7. Dalším zástupcem jednofázových detektorů je YOLO (You Only Look Once). Jedná se o detektor sestávající z jedné neuronové sítě, která přímo ze vstupního obrazu predikuje rámečky kolem objektů i pravděpodobnosti příslušných tříd. První verze byla představena v roce 2016 [24], její architektura je zobrazena ve spodní části obrázku 2.7. Experimenty ukázaly, že hlavním problémem metody je nepřesná lokalizace. O rok později byla zveřejněná druhá verze [25], která představila několik nových konceptů ke zvýšení přesnosti a také rychlosti. Autoři mimo jiné navrhli novou páteřní síť pojmenovanou Darknet-19, která potřebuje pro zpracování obrazu méně operací a přesto je přesnější. V roce 2018 byla představena třetí verze. Ta používá větší páteřní síť Darknet-53 (53 konvolučních vrstev místo 19 ve druhé verzi) inspirovanou ResNet architekturou, což způsobilo zpomalení metody, ale zvýšilo přesnost. Nejdůležitější vlastností YOLOv3 je, že detekuje ze třech různých měřítek příznaků. Také už nepoužívá při predikci třídy objektu softmax funkci, která vrací třídu s největší pravděpodobností, ale používá logistickou regresi pro určení pravděpodobností a pomocí rozhodné hodnoty předpoví, do kterých tříd objekt patří. To umožňuje klasifikaci objektů z komplexnějších datových sad, kde jeden objekt může spadat do více tříd. V roce 2020 byla představena poslední, čtvrtá verze YOLOv4 [26]. Opět byla použita nová páteřní síť nazvaná CSPDarknet53 [27]. Autoři této verze experimentovali s různými metodami pro zvýšení úspěšnosti jako je augmentace dat či použití Mish aktivační funkce. Několik metod adaptovali do nové architektury YOLOv4 a dosáhli tím zvýšení rychlosti i přesnosti. Podle autorů se jedná o aktuálně nejrychlejší (FPS) a nejpřesnější detektor objektů nad datovou sadou MS COCO [17].



Obrázek 2.7: Porovnání architektury YOLO a SSD detektoru [23].

Tabulka 2.1: Porovnání výsledků metod pro detekci objektu nad datovou sadou MS COCO. Použité metriky: AP, AP_{50} , AP_{75} skóre (%). AP_S:AP pro malé objekty, AP_M:AP pro střední objekty, AP_L:AP pro velké objekty [12].

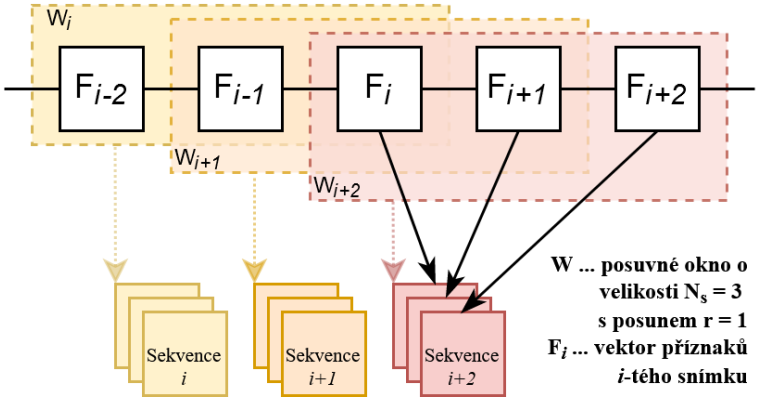
Metoda	Data	Páteřní síť	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Fast R-CNN	train	VGG-16	19,7	35,9	-	-	-	-
Faster R-CNN	trainval	VGG-16	21,9	42,7	-	-	-	-
YOLOv3	trainval35k	DarkNet-53	33,0	57,9	34,4	18,3	35,4	41,9
YOLOv2	trainval35k	DarkNet-19	21,6	44,0	19,2	5,0	22,4	35,5
SSD321	trainval35k	ResNet-101	28,0	46,1	29,2	7,4	28,1	47,6
SSD513	trainval35k	ResNet-101	31,2	50,4	33,3	10,2	34,5	49,8

2.2 Reprezentace dat

Nalezený objekt v obraze je potřeba vhodným způsobem reprezentovat pro následnou analýzu. Pro popis akcí prováděných člověkem ve videu musí být zachycen jeho pohyb v relevantním časovém úseku. Pokud video obsahuje více objektů zájmů, např. záznam z bezpečnostních kamer, je dalším problémem jejich sledování. Je tedy potřeba v po sobě jdoucích snímcích detekované objekty správně rozlišit tak, aby se výsledná sekvence týkala jednoho člověka. Tato práce je však zaměřena primárně na analýzu chování řidiče vozidla, kde tento problém nenastává, a proto zde nebude dále rozebírán.

Nejjednodušší metodou pro zachycení časové osy, která je ale zároveň často používaná a účinná, je vytvoření sekvence z po sobě jdoucích snímků videa. Z každého snímku videa se získají vektory příznaků popisující sledovaný objekt a ty se sestaví za sebe tak, aby vytvořily vektor popisující akci prováděnou objektem v časovém úseku. Tento vektor má tedy navíc časovou dimenzi. Video je možné rozdělit na nepřekrývající se úseky, což ale může způsobit ztrátu informace. Při modelování vektorů příznaků pro popis a následnou analýzu lidského chování se využívá techniky posuvného okna (sliding window), která je znázorněna na obrázku 2.8. Akce prováděná v i-tém snímku (S_i) je pak popsána sekvencí příznaků z po sobě jdoucích snímků S_{i-N_S-1} až S_i, kde N_S je předem definovaná velikost okna. Obecně se u metody posuvného okna definuje ještě druhý parametr, kterým je posun r. Pro výše popsaný účel získání sekvencí pro každý snímek videa je posun $r = 1$.

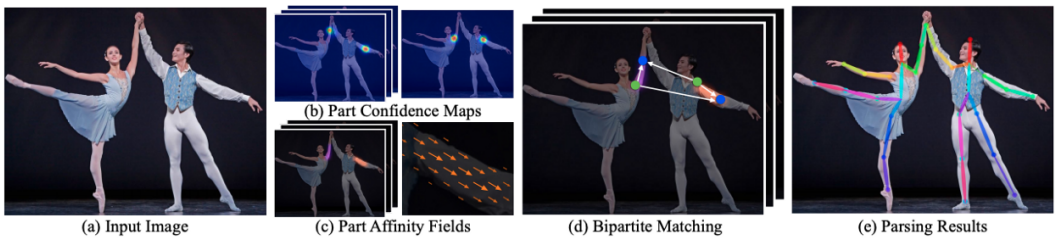
Jako vektor příznaků popisující nalezený objekt v jednom snímku může být zvolen např. některý z deskriptorů popsaný v předchozí podkapitole (HOG, HAAR či výstup CNN). Pro popis člověka za účelem analýzy prováděné akce (pohybu) je však vhodnější zvolit reprezentaci, která přímo popisuje body zájmu. Pro účely sledování celého těla a jeho pohybu se používá reprezentace kostry, konkrétně zde uvedu algoritmus OpenPose, pro popis tváře a výrazu pak orientační body obličeje (facial landmarks).



Obrázek 2.8: Znázornění metody posuvného okna (sliding window)

2.2.1 OpenPose

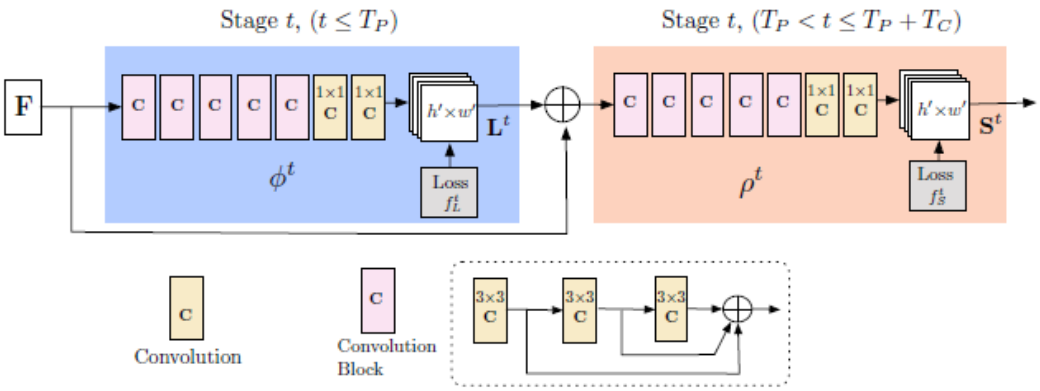
OpenPose je algoritmus pro odhad pozice lidí z obrazu v reálném čase, který byl vyvinutý výzkumnými pracovníky z Carnegie Mellon University. Jedná se o open-source knihovnu napsanou v jazyce C++ využívající framework Caffe. Dostupné je také API umožňující používání většinu funkcí z Pythonu. Poprvé byl představen v práci publikované v roce 2016 [28]. V roce 2018 publikovali vylepšenou verzi algoritmu [29], která zvyšuje přesnost a rychlost, ale základní myšlenka a postup zůstaly stejné. Zde se budu zabývat novější verzí algoritmu. Běžné přístupy nejprve v obraze detekují každého člověka a následně provádí odhad pozice u každého z nich, tak jak to obecně uvádím i v této práci. Hlavním problémem je, že s narůstajícím počtem lidí v obraze narůstá proporcionálně také časová složitost, protože pro každého člověka je spuštěn odhad pozice. OpenPose představuje přístup, který ze vstupního RGB obrazu odhadne pózy lidí, kteří se v něm vyskytují v čase nezávislém na jejich počtu. Podle autorů je robustní vzhledem k problému chybějící detekce a zároveň efektivní. I když algoritmus provede také detekci osob, rozhodla jsem se jej zařadit do této části práce, protože jeho hlavním účelem je odhad pózy člověka a její reprezentace.



Obrázek 2.9: Postup algoritmu OpenPose [29].

Na obrázku 2.9 je znázorněný postup OpenPose. Systém ze vstupního RGB obrazu generuje 2D lokace klíčových bodů lidského těla pro všechny osoby (obrázek 2.9e). Obraz je nejprve zpracován CNN sítí, která generuje vektory příznaků F. V publikaci je použita síť VGG-19. Vektory příznaků

jsou použity jako vstup do první části, která iterativním procesem v několika fázích generuje částečné afinní pole (PAF) (obrázek 2.9c). V každé další fázi t je predikce té předchozí (L^{t-1}) spojená s vektorem příznaků F a použita ke generování přesnější predikce. Po provedení T_P iterací jsou obdobným iteračním procesem generovány tzv. confidence maps S (obrázek 2.9b). Využívá se při tom PAF z poslední fáze předchozí části L^{T_P}, opět spojený s odpovídajícím vektorem příznaků F. V každé další fázi se spojí výsledek té předchozí s vektorem příznaků i s L^{T_P}. Celkem se provede T_C iterací. Struktura těchto částí je znázorněna na obrázku 2.10. Následně jsou nalezené body v obraze biparitním párováním pospojovány tak, aby vytvořily pózu (kostru) pro každého člověka. Částečné afinní pole (PAF) je soubor 2D vektorů, které kódují lokaci a orientaci končetin v obraze.

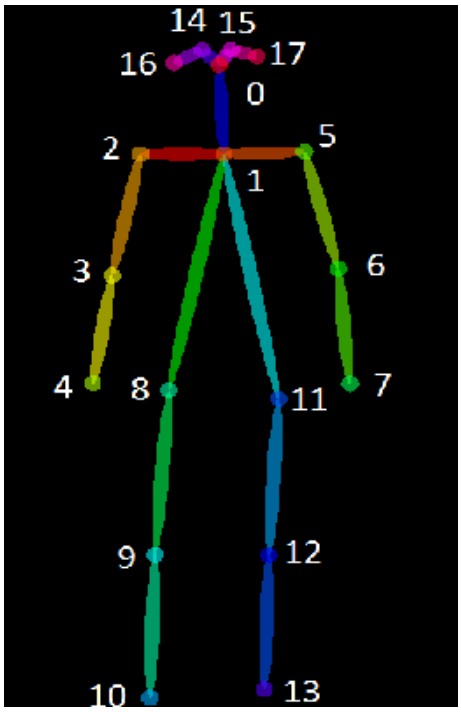


Obrázek 2.10: Struktura OpenPose pro predikci PAFs (modrá část, výstupem je L^t pro každou fázi t) a confidence maps (oranžová část, výstupem je S^t pro každou fázi t) [29].

Confidence map je reprezentace pravděpodobnosti, že se určitá část těla (bod) nachází v daném místě obrazu. Celkový počet map odpovídá počtu detekovaných bodů končetin. Za končetinu je zde považována část těla, která se kóduje mezi dvěma body. Jejich počet a umístění se liší v závislosti na použitých trénovacích datech. Např. pro datovou sadu MS COCO je těchto bodů celkem 18 (13 pro tělo a 5 pro hlavu) viz obrázek 2.11, celkem tedy 17 končetin.

2.2.2 Orientační body obličeje

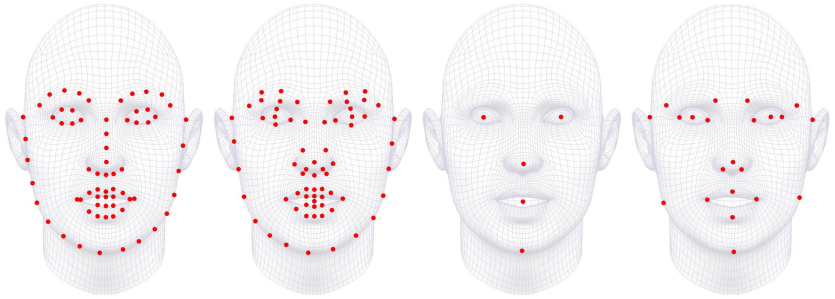
Orientační body obličeje mají za úkol co nejpřesněji popsat důležité části tváře tak, aby utvářely jeho ucelenou reprezentaci. Jejich počet a rozmístění záleží na použité metodě a datech, na kterých byla natrénovaná. Na obrázku 2.12 je znázorněno rozložení definované v osmi různých datových sadách. Každá používaná reprezentace popisuje základní části tváře, kterými jsou oči, nos a ústa. Většina pak také definuje body pro obočí a čelist. Predikce orientačních bodů obličeje se uplatňuje v mnoha aplikacích zaměřených na tvář, např. při analýze výrazu, rozpoznávání lidí či pohlaví, u detekce mrkání, nebo u populárních obličejových filtrů. Díky tomu existuje mnoho prací a metod zaměřených na tuto problematiku. Základem každé z nich je správně detekovaná tvář. Zde uvedu dvě práce a jejich navrhované metody.



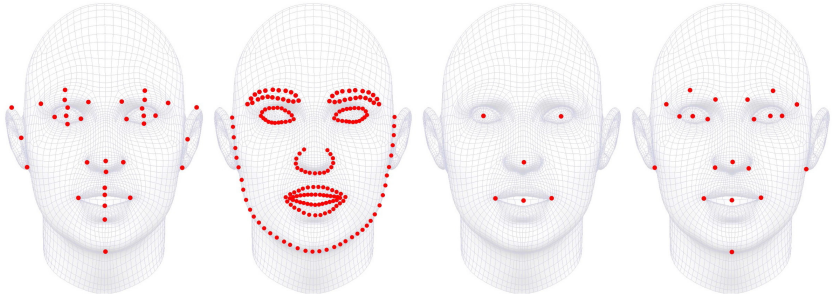
Obrázek 2.11: Klíčové body reprezentující postavu člověka v datové sadě MS COCO [30].

1. **Ensemble of Regression Trees (ERT)** Kazemi a Sullivan v roce 2014 představili metodu pro lokalizaci 2D obličejových orientačních bodů založenou na kaskádě regresních stromů, která dosahuje rychlosti v jednotkách milisekund [32]. V této práci ji uvádím především z důvodu, že je využívaná knihovnou Dlib [33]. Metoda lokalizuje body v obraze přímo na základě intenzity pixelů, nevyužívá tedy žádné extrakce příznaků. Základem metody je kaskáda regresních funkcí, které se při trénování učí pomocí boostingu gradientních stromů. Počet kaskád v publikaci uvádí celkem deset, přičemž každá z nich se skládá z pěti set slabých regresorů. Při predikci pozice orientačních bodů tváře se nejprve nastaví jejich výchozí pozice. Doporučená je průměrná pozice z trénování, která se nastaví do středu detekovaného obličeje. Následně postupně prochází kaskádou, přičemž každá fáze pozici aktualizuje. Postupná úprava pozic je znázorněna na obrázku 2.13.

2. **Face Alignment Network (FAN)** Tzimiropoulos a Bulat publikovali v roce 2017 práci, ve které představili predikci orientačních bodů obličeje pomocí konvoluční neuronové sítě nazvané Face Alignment Network (FAN) [34]. Pro konstrukci použili v té době jednu z nejmodernějších architektur pro odhad pózy člověka, Hour-Glass (HG) [35]. Konkrétně použili soustavu čtyř HG sítí, ve kterých zaměnili hlavní stavební blok. Využili hierarchický, paralelní a víceúrovňový blok, který sami navrhli ve své předchozí práci [36]. S jeho použitím dosáhla síť lepších výsledků při zachování stejného počtu parametrů sítě. Architektura použitého stavebního bloku a výsledné FAN sítě je na obrázku 2.14.

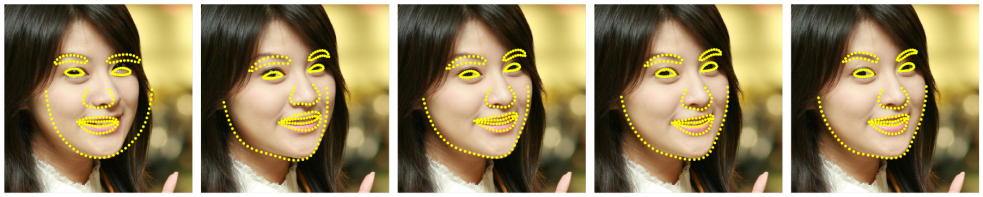


(a) MultiPIE/IBUG (b) XM2VTS (c) FRGC-V2 (d) AR



(e) LFPW (f) HELEN (g) AFW (h) AFLW

Obrázek 2.12: Rozložení orientačních bodů popisujících tvář v osmi různých datových sadách. Každý z nich definuje jiný počet i rozmístění [31].

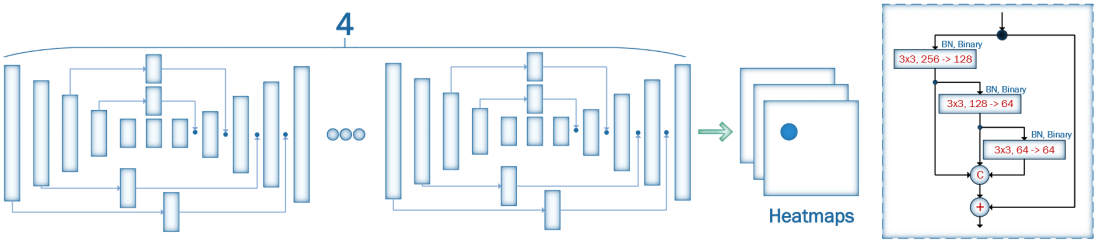


(a) $T = 0$ (b) $T = 1$ (c) $T = 2$ (d) $T = 3$ (e) $T = 10$

Obrázek 2.13: Odhady pozice orientačních bodů obličeje v různých fázích (T) kaskády regresních stromů [32].

V rámci své práce také vytvořili síť se stejnou architekturou jako FAN, kterou použili pro generování 3D orientačních bodů. Jako vstup použili vstupní RGB obrázek spolu s 2D orientačními body. Přesněji řečeno vstupní RGB kanály byly rozšířeny o dalších 68, jeden kanál pro každý 2D orientační bod. Tyto kanály obsahovaly 2D Gaussovo rozložení se směrodatnou odchylkou 1 a střední hodnotou na pozici příslušného bodu. Tuto síť nazvali 2D-to-3D FAN. Pro trénování použili sadu dat 300-W-LP, která má jak 2D, tak 3D anotace orientačních bodů pro každou tvář. 3D orientační body produkované touto sítí jsou ve skutečnosti pouze 2D projekcí. Pro predikci třetí dimenze (z) zkonstruovali další síť přidáním ResNet-152 sítě. Ta ze vstupního RGB obrazu a map produkovaných 2D-to-3D FAN sítí predikuje z dimenzi jednotlivých orientačních bodů.

Pomocí natrénované 2D-to-3D FAN sítě poté vytvořili soubor dat LS3D-W, který obsahuje



Obrázek 2.14: Face Alignment Network (FAN) zkonstruovaná ze čtyř HG sítí. Modré obdélníky představují základní stavební blok, jehož architektura je v pravé části obrázku [34].

Obrázek 2.15: Ukázka predikce orientačních bodů obličeje sítí 2D-FAN (a) a 3D-FAN (b). Hlavní rozdíl je vidět na oblasti čelisti, která u 2D predikce tváře pod úhlem kopíruje viditelnou část tváře místo reálné linie [34].

230 tisíc snímků z různých obličejových datových sad spolu s 3D anotacemi orientačních bodů (2D projekce). Ten použili pro natrénování 3D-FAN sítě, která se ukázala jako spolehlivá pro snímky tváře pod různým úhlem (-90°až +90°). Na obrázku 2.15 je ukázka některých predikcí sítí 2D-FAN a 3D-FAN. Hlavní rozdíl je viditelný u tváří pod větším úhlem, kdy část obličeje není viditelná. 2D anotace v takovém případě kopíruje viditelnou linii tváře, zatímco 3D predikuje reálnou podobu. Prezentované metody spolu s natrénovanými sítěmi a ukázkovými kódy jsou dostupné na stránkách autora a zároveň implementované jako modul v pythonu, což umožňuje jejich jednoduché použití.

2.3 Analýza dat

Pro účely detekce anomálií bylo představeno mnoho metod. Např. metody založené na k-NN algoritmu, nebo Skrytém Markovu Modelu (HMM). Níže v této části budou popsány metody využívající

rekurentní neuronové sítě (RNN) a variační autoencodér (VAE). Základem klasifikačních metod, které jsou založené na učení s učitelem (supervised), je dostatečně velký soubor dat s vyváženým poměrem klasifikovaných tříd. Anomální data jsou však ze své podstaty ojedinělá a nelze získat data pro všechny možné případy anomálií. Konkrétně u sledování chování řidiče vozidla nejsou dostupná reálná videa zachycující např. vážné zdravotní problémy či usínání za běžných podmínek provozu. Pro detekci anomálií se proto často využívají metody s částečným učitelem (semi-supervised). Těm se při trénování dodají data obsahující pouze normální chování. Pokud se data při evaluaci vymykají těmto naučeným vzorům, jsou označena za anomální. Stále je ovšem potřeba co největšího objemu trénovacích dat, aby se síť naučila normální vzory a dokázala je rozpoznat od těch anomálních u neznámých dat při evaluaci. Velký důraz je přitom kladen na to, aby se eliminovaly falešně pozitivní detekce.

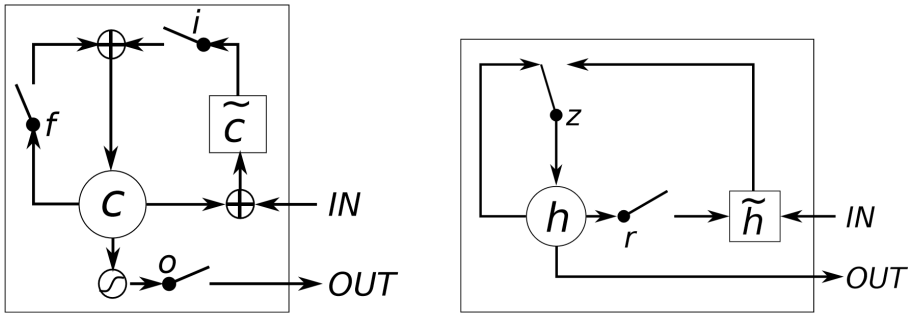
Za účelem zpracování sekvence dat se používají rekurentní neuronové sítě (RNN). Sekvence je složena z vektorů příznaků v určitém počtu po sobě jdoucích snímků a v případě analýzy chování popisuje prováděnou akci. Pro účely detekce anomálií se RNN využívají jako predikční sítě natrénované pouze na normálních datech. Při testování by pak pro anomální data měly predikovat s velkou chybou. Další využívanou metodou je použití autoenkodéru, nejčastěji variačního (VAE). Jeden z možných způsobů použití je, že se snaží co nejpřesněji rekonstruovat svůj vstup, přičemž se opět trénuje pouze na datech neobsahujících anomálie. Při evaluaci by pak měl mít pro anomální data vysokou rekonstrukční chybu. Další z možností použití VAE pro detekci anomálií je spolu s SVM klasifikátorem s jednou třídou. V takovém případě je výstup z kodéru natrénovaného VAE použitý jako vstup do SVM, který detekuje odlehlé hodnoty (anomálie).

2.3.1 Rekurentní neuronové sítě (RNN)

Myšlenkou za RNN sítěmi je zpracování sekvenčních informací. V tradiční neuronové síti předpokládáme, že všechny vstupy (a výstupy) jsou na sobě nezávislé. RNN vykonávají stejnou úlohu pro každý prvek sekvence, přičemž výstup je závislý na předchozích výpočtech (vstupech). Toho je docíleno skrytou vrstvou sítě, která počítá výstup na základě předchozího stavu a aktuálního vstupu. Teoreticky mohou využívat informace v libovolně dlouhých sekvencích. V praxi jsou však omezeny pouze na pár předchozích kroků v závislosti na architektuře sítě. Problém, který to způsobuje, se označuje jako mizející gradient, při kterém se postupně vytrácí informace o předchozích vstupech [37].

Speciálním případem RNN jsou LSTM (Long Short-Term Memory) a GRU (Gated Recurrent Unit), které částečně eliminují problém mizejícího gradientu a díky tomu dokáží brát v potaz delší sekvence dat. Využívají při tom složitějších mechanizmů pro výpočet výstupní hodnoty buňky. LSTM buňky obsahují vnitřní paměť a tři brány: vstupní i, zapomínající f a výstupní o. Vstupní brána určuje, jak velká část nově vypočteného skrytého stavu buňky bude použita. Obdobně zapomínající brána definuje jaká část z předešlého stavu paměti bude použita. Z těchto dvou hodnot

je určen nový stav vnitřní paměti. Výstupní brána poté určuje, jak velká část této paměti bude předána dále. GRU buňky mají pouze dvě brány: aktualizační z a resetovací r. Resetovací buňka určuje v jakém poměru zkombinovat předchozí stav s aktuálním vstupem, zatímco aktualizační definuje jakou část předchozího stavu zachovat [37]. Ilustrace struktury buněk je na obrázku 2.16. GRU oproti LSTM nemají přidanou vnitřní paměť a kvůli absenci výstupní brány předávají dál celý svůj stav. Experimenty uvedené v práci [37] ukázaly, že sítě s těmito typy buněk dosahují lepších výsledků než klasické RNN. Neprojevily se však výraznější rozdíly mezi použitím GRU a LSTM.



(a) Long Short-Term Memory (b) Gated Recurrent Unit

Obrázek 2.16: (a) Struktura LSTM buňky. i, f, o představují vstupní, zapomínající a výstupní bránu. c značí vnitřní paměť a $\widetilde{c}$ nový obsah paměti. (b) Struktura GRU buňky. r představuje resetovací bránu a z aktualizační. h značí aktivaci (vnitřní stav) a $\widetilde{h}$ kandidáta na aktivaci [37].

Sítě složené z RNN vrstev se často používají pro predikci blízké budoucnosti časové řady. V takovém případě je síť složená z části, která vstupní sekvenci zpracuje a zakóduje do definovaného prostoru. Následně se z této reprezentace predikuje vektor příznaků popisující stav v blízké budoucnosti. Při trénování se takové sítí dávají za vstup sekvence vektorů příznaků a požadovaný výsledek, tedy vektor příznaků popisující stav v požadovaném okamžiku následujícím zpracovávanou sekvenci. Síť se pomocí zpětné propagace chyby, která udává rozdíl mezi generovaným a požadovaným výsledkem učí co nejlépe predikovat budoucnost. U detekce anomálií se využívá předpokladu, že pokud se takováto síť trénuje pouze na normálních datech, bude při evaluaci pro anomální data predikovat data s velkou odchylkou od skutečných. Při takovém přístupu je nutné při trénování použít dostatečné množství normálních dat. Zároveň nesmí dojít k přeučení sítě, protože v takovém případě by se naučila správně predikovat pouze trénovací data. Výsledný model by měl umět rozpoznat určité vzory ze vstupních dat a na základě nich predikovat. Je potřeba vhodně zvolit parametry, jako jsou velikost a hloubka sítě, délka sekvence, frekvence vzorkování dat z časové řady, či vzdálenost predikované budoucnosti. Parametry záleží na dimenzi a povaze analyzovaných dat, účelu aplikace, jejím požadovaném použití apod.

2.3.2 Variační autoenkodér (VAE)

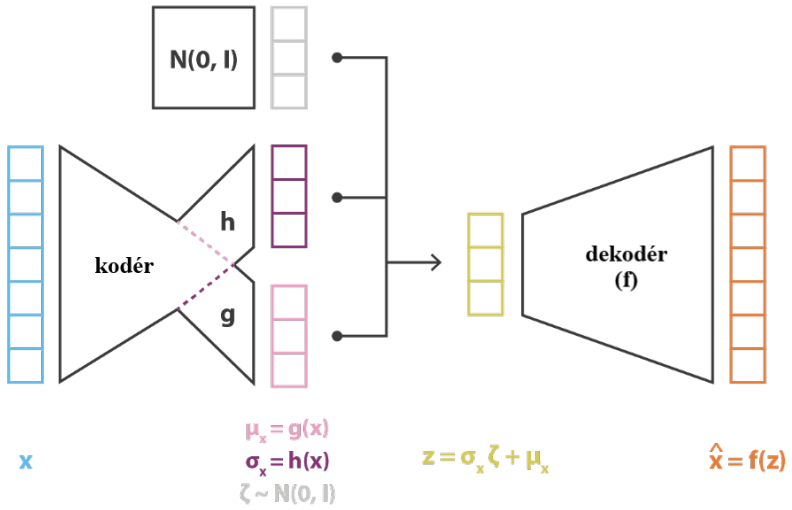
Autoenkodér je typ architektury neuronové sítě, která sestává ze dvou částí: kodér a dekodér. Kodér provádí operaci redukce dimenze vstupních dat do tzv. latentního prostoru a dekodér je rekonstruuje zpět. U klasického autoenkodéru, který kóduje vstupní data jako jeden bod z latentního prostoru dochází k tomu, že rozdělení tohoto prostoru není ideální. Vznikají oddělené shluky a dochází tím v podstatě k přeučení sítě, protože dekodér pro data z latentního prostoru mimo tyto body a shluky bude produkovat nesmyslná data. Variační autoenkodér provádí kódování vstupních dat místo jednoho bodu jako rozložení v latentním prostoru. Výstupem kodéru jsou dva vektory: střední hodnota μ a směrodatná odchylka σ. Pro účely dekódování se provádí vzorkování z takto definovaného rozložení. Proces vzorkování musí být proveden způsobem, který umožňuje zpětnou propagaci chyby při trénování. Neprovádí se tedy přímo z definovaného rozložení jako

$$z \sim N(\mu_x, \sigma_x), \tag{2.5}$$

ale využívá se faktu, že náhodná proměnná z je z normálního rozložení a může být tedy vyjádřena jako

$$z = \sigma_x \zeta + \mu_x, \zeta \sim N(0, I) \tag{2.6}$$

Díky tomu je umožněna zpětná propagace chyby kodéru při trénování. Na obrázku 2.17 je ilustrace kompletního modelu VAE vyznačující jednotlivé části se stejným značením jako je uváděno výše. Stále však může docházet k vytvoření oddělených shluků. Proto se do chyby modelu přidává



Obrázek 2.17: Ilustrace modelu variačního autoenkodéru. Popis a význam použitých proměnných je v části 2.3.2 - VAE [38].

Kullback-Leiblerova (KL) divergence, která udává jak moc jsou dvě pravděpodobnostní rozložení rozdílné. Pro účely VAE se počítá mezi získaným rozložením z kodéru a standardním Gaussovým rozložením. Díky tomu, že obě rozložení jsou normální, může být KL divergence počítána v uzavřené formě. Výpočet KL divergence u VAE má formu

$$\mathrm{KL}_x = \sum_{j=1}^{J} \frac{1}{2} \left[\sigma_x^2 + \mu_x^2 - log(\sigma_x^2) - 1 \right], \tag{2.7}$$

kde J značí dimenzi latentního vektoru. Odvození rovnice (2.7) je dostupné z [39]. Regularizace přidáním KL divergence do chyby sítě zajišťuje, že kódovaná data nebudou v latentním prostoru příliš vzdálené [38, 40]. Díky tomu budou i data mezi těmito prostory po dekódování smysluplná jak je znázorněno na obrázku 2.18, kde je také ilustrace latentního prostoru VAE rozděleného kontinuálně a takového, kde jsou vytvořeny oddělené shluky.

Kompletní model VAE se trénuje pomocí zpětné propagace chyby modelu, která se skládá z rekonstrukční chyby a výše popsané KL divergence. Rekonstrukční chyba udává rozdíl mezi původními a dekódovanými daty. Pro její výpočet se používá např. střední kvadratická odchylka. Přesná architektura a parametry modelu VAE záleží na jeho požadované aplikaci. Samotný model sítě (kodéru a dekodéru) může být složen z různých typů vrstev. Např. pro zpracování obrazu se využívá CNN, při práci se sekvencemi RNN apod.

Obrázek 2.18: Ilustrace možného rozdělení latentního prostoru bez použití regularizace (vlevo) a s použitím regularizace pomocí KL divergence (vpravo). V levé části obrázku je znázorněna nesmyslnost znaku dekódovaného z prostoru mezi shluky, zatímco v pravé části je znak dekódovaný ze středu prostoru sloučením těch okolních [38].

Kapitola 3

Související práce

Detekcí anomálií pomocí hlubokého učení se zabývá v posledních letech mnoho prací. Studie [41] z roku 2019 obsahuje souhrn prací a aplikací na toto téma. Nejčastěji jsou zmiňované metody využívající CNN, RNN (LSTM, GRU) a autoenkodéry. Studie [42] ze začátku roku 2021 přestavuje souhrn prací zabývající se detekcí a lokalizací anomálií ve videu, nejčastěji z bezpečnostních kamer, pomocí hlubokého učení. Pro účely zpracování videa jsou zmiňované hlavně CNN, konkrétně VGG [15] a YOLO [43] architektury. Často jsou v kombinaci s LSTM sítěmi, nebo autoenkodéry. Pro následnou detekci anomálií je nejvíce používáno generativních a prediktivních modelů. Studie také představuje souhrn dostupných datových sad a popisuje metodiky používané pro hodnocení úspěšnosti. V [44] autoři navrhli metodu pro detekci a lokalizaci anomálního lidského chování z videa z bezpečnostních kamer na základě trajektorie kostry člověka. Každá detekovaná postava je reprezentovaná pózou (kostrou) nezávislou na měřítku a globální pozicí ve snímku. Metoda využívá kodér a dva dekodéry z nichž jeden predikuje budoucí snímky a druhý rekonstruuje vstup. Trénování probíhá pouze na normálních datech za účelem následné detekce anomálií jako sekvencí vymykající se naučeným vzorům. V [45] autoři přestavili metodu založenou na autoenkodéru. Obrazová data jsou zpracována CNN vrstvami a následně je pomocí konvolučních LSTM (ConvLSTM) vrstev zachycena časová osa. Opět využívají techniky trénování pouze na normálních datech. Autoři uvádějí možnou rychlost až 140 zpracovaných snímků za sekundu. V publikaci [46] použili pro detekci anomálií metodu založenou na predikci budoucího snímku. Architektura použité sítě pro predikci vychází z U-Net [47]. V práci [48] použili pro detekci anomálií ve videu z bezpečnostních kamer metodu založenou na oboustranné predikci, kdy je snímek predikován na základě sekvence předchozích a následujících snímků. Architektura použité sítě je opět založená na U-Net [47]. Metoda překonala většinu v té době nejúspěšnějších metod nad několika soubory dat.

Detekcí anomálií v chování řidiče vozidla se zabývají autoři práce [49]. Zároveň v něm představují veřejně přístupnou datovou sadu Driver Anomaly Detection (DAD), která obsahuje videa zachycující řidiče vozidla hloubkovou a infračervenou kamerou ze dvou pozic. První kamera snímá řidiče z palubní desky zatímco druhá je umístěná nad řidičem a směřuje na jeho ruce. Navrhovaná metoda

pro detekci anomálního chování je založená na kontrastním přístupu k učení, kdy se při trénování maximalizuje vzdálenost mezi normálními a anomálními daty (v latentním prostoru). Autoři v práci uvádějí porovnání výsledků při použití několika architektur sítě. Nejlepších výsledků dosáhla síť ResNet [13] s 18 vrstvami. Další práce sledující chování řidiče se převážně zaměřují pouze na určitou problematiku, nejčastěji na detekci únavy a usínání. Metodu založenou na sledování pohybu hlavy pomocí detekce očí a úst s využitím YOLO sítě a klasifikace pomocí SVM představili autoři v [50]. Procento uzavření očních víček v průběhu času (PERCLOS) využili pro detekci usínání v práci [51]. Pro detekci obličeje a očí využili autoři Haarovy příznaky spolu s AdaBoost algoritmem [3]. Práce [52] se zabývá detekcí zívání a mluvení řidiče vozidla pomocí sledování oblasti úst. V práci [53] se zabývají autoři detekcí usínání řidiče. Metoda je založená na detekci očí a úst s následným použitím PCA a klasifikací stavu pomocí SVM.

Několik studií a aplikací bylo také zaměřeno na analýzu stavu řidiče za použití speciálních senzorů zachycující signály jako jsou EEG, ECG či teplota kůže. Nevýhodou těchto metod je však potřeba umístění senzorů nejčastěji na hlavě řidiče, což pro něj může být nepříjemné a rozptylující. Tyto metody jsou shrnuté ve studii [54], kde jsou popsány také asistenční funkce používané ve vozidlech pro analýzu stylu řízení a hlídání okolí. Např. dnes běžné hlídání jízdních pruhů, sledování vzdálenosti od okolních vozidel apod.

Kapitola 4

Vlastní implementace

Na základě teoretických znalostí jsem implementovala vlastní detektor anomálií určený pro sledování stavu řidiče vozidla. V této kapitole se budu zabývat nejprve použitými daty, následně volbou postupů v jednotlivých částech aplikace se zdůvodněním výběru. Na závěr shrnu navrženou metodu, dosažené výsledky, její výhody a nedostatky. Jedním z aspektů důležitých pro výslednou aplikaci a tím i volbu metod je potřebný čas na zpracování. Všechny časy a výsledky prezentované v této části byly měřeny na zařízení s grafickou kartou Nvidia GeForce GTX 1050 4GB.

4.1 Použitá datová sada

Pro tuto práci jsem dostala přístup k videím, které byly natočeny pro účely studií na podobná témata. Jedná se o videa zachycující tři řidiče natáčené pomocí několika typů kamer s jejich různým umístěním ve vozidle. Pro svou práci jsem z nich vybrala taková, ve kterých kamera snímá řidiče pod co nejmenším úhlem tak, aby byla analýza jeho tváře co nejpřesnější a nejspolehlivější. Tomu odpovídají záběry z kamer umístěných na vnitřním zpětném zrcátku. Tato videa jsem dále rozdělila na normální a anomální, kdy normální neobsahují žádné anomálie a jsou určené k trénování. Všechna videa, i anomální, byla natočena za jízdy. Anomálie obsažené v testovacích datech jsou různého typu, např. kašel, spánek, nebo telefonování. Z vybraných normálních videí jsem odstranila ta s výrazně špatným osvětlením, protože v takových případech byla detekce obličeje a orientačních bodů špatná a pro trénování tedy nevhodná. Všechna videa jsou na začátku zpracovávání otočena tak, aby byl řidič na levé straně z pohledu kamery. Ukázky z videí jsou na obrázku 4.1. Výsledná trénovací datová sada obsahuje 10 videí zachycujících pouze normální aktivity. Dohromady se jedná o necelé dvě hodiny videa s FPS 30, což je bez mála 287 tisíc snímků. Z toho dvě videa v celkové délce 27 minut zachycují řidiče ve slunečních brýlích.

Pro účely testování slouží 7 videí, která dohromady dávají bez mála 50 minut. Z toho jedno devíti minutové video zachycuje řidiče ve slunečních brýlích. Obsažené anomálie jsou nasimulované za jízdy. Pro účely hodnocení testovaných metod jsem je rozdělila do pěti tříd, které jsou uvedené

Obrázek 4.1: Ukázky z trénovací sady dat.

Tabulka 4.1: Třídy anomálií a jejich procentuální zastoupení v testovacích datech.

Třída	Popis anomálie	Zastoupení v testovacích datech (%)
1	Spánek - sklopená hlava dopředu	9,4
2	Spánek - zakloněná hlava dozadu	8,2
3	Zívání	4,5
4	Kašel (dušení se)	4,0
5	Telefonování	4,8
Celkem anomálií		**30,9**

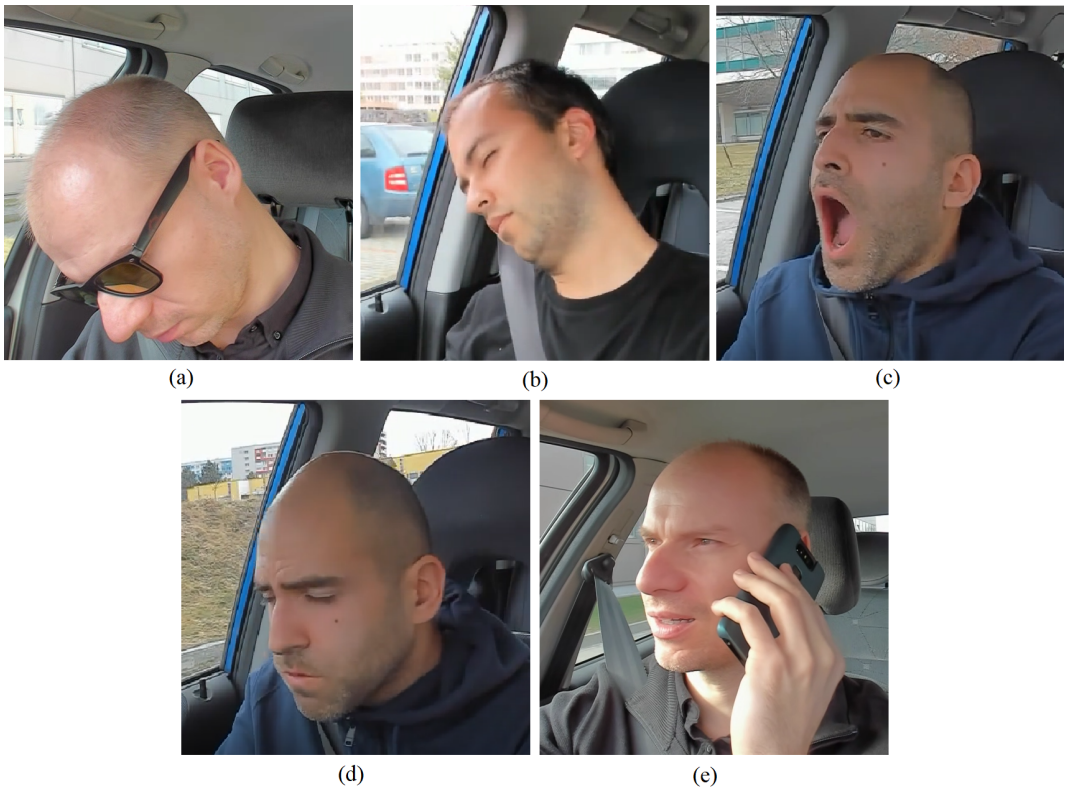
spolu s jejich četností v testovacích datech v tabulce 4.1. Ukázky snímků obsahujících jednotlivé anomálie jsou na obrázku 4.2. Ke všem videím jsem pro účely hodnocení metod vytvořila soubory s anotacemi, které obsahují vždy začátek a konec (číslo snímku) anomálie a její typ (třídu).

4.2 Volba postupů

Pro každou část vyvíjeného detektoru anomálií jsem implementovala a testovala několik metod. Níže uvádím jejich popis, výhody a nevýhody. Závěrem každé části je pak zvolená metoda a případné problémy, jejich řešení a implementační detaily.

4.2.1 Detekce člověka

Stav řidiče vozidla nejlépe odráží jeho tvář a pozice hlavy. Proto ze vstupního obrazu detekuji jeho obličej. Tato část je kritická pro celou aplikaci. Pokud není detekovaná tvář nemůže proběhnout



Obrázek 4.2: Ukázky snímků obsahujících detekované anomálie: (a) spánek - sklopená hlava, (b) spánek - zakloněná hlava, (c) zívání, (d) kašel, (e) telefonování.

analýza. V případě špatné predikce pozice obličeje jsou výsledná data nesmyslná. Při výběru metody jsem se tedy zaměřila především na úspěšnost detekce. Jak jsem uvedla v části popisující použitá data, kamera snímající řidiče je umístěná na vnitřním zpětném zrcátku. Řidič je tedy snímán pod úhlem a při řízení pohybuje hlavou především do stran. Je tedy potřeba, aby byl detektor schopný správně nalézt obličej i pod značným úhlem. Vzhledem ke snaze umožnit běh aplikace v reálném čase byl dalším kritériem potřebný čas. Za tímto účelem jsem vyzkoušela několik metod dostupných v knihovnách Dlib [55], Cvlib [56] a OpenCV [57]. Níže uvádím stručný popis implementovaných metod, jejich výhody a nevýhody. Dále vždy uvádím průměrný čas potřebný na zpracování jednoho snímku, který jsem měřila včetně všech nezbytných operací pro předzpracování obrazu a následného výběru řidiče ze všech detekovaných tváří. Příklady pozic hlavy, kde jednotlivé metody (vyjma OpenCV Haar detektoru) již nedetekovaly tvář jsou na obrázku 4.4.

1. **OpenCV Haar detektor**

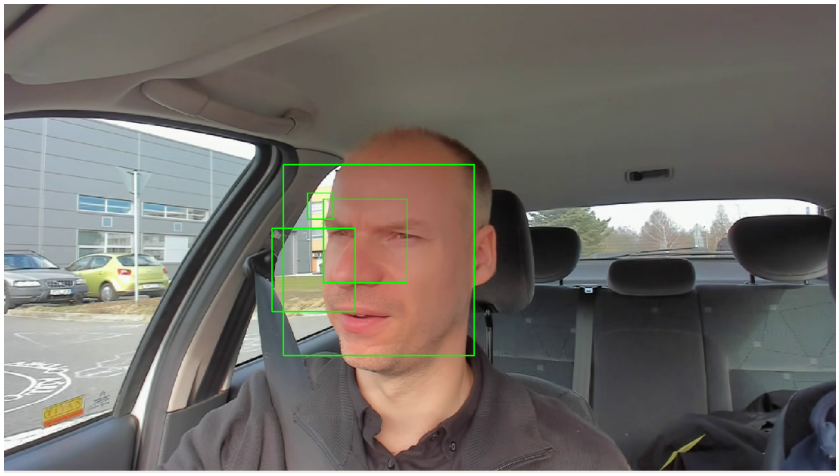
   Knihovna OpenCV obsahuje metodu pro načtení kaskádového klasifikátoru z XML souboru. Současně nabízí několik modelů (souborů) pro kaskádový klasifikátor založený na Haarových příznacích podle práce Viola a Jonese [3]. Pro svou práci jsem otestovala základní model pro detekci tváří. Hlavním problémem této metody je mnoho falešných a vícenásobných detekcí.

Příklad výstupu tohoto detektoru s několika detekcemi je na obrázku 4.3. Následný výběr správné detekce ztěžuje také fakt, že metoda neudává na kolik procent si je jistá danou detekcí, jako je tomu u ostatních metod. Detekce je úspěšná pouze pro tvář namířenou přímo do kamery, nebo pod minimálním úhlem. Vzhledem k pozici kamery, která snímá pohybujícího se řidiče mírně ze strany, se tato metoda ukázala jako naprosto nevyhovující, protože ve většině snímků tvář nedetekovala vůbec, nebo špatně.

Průměrný čas: 152 ms

Výhody: jednoduchost modelu

Nevýhody: nepřesná nebo špatná detekce i pod malým úhlem, velký počet falešných detekcí, vyšší čas



Obrázek 4.3: Příklad výstupu OpenCV Haar detektoru s několika falešnými a vícenásobnými detekcemi pro jednu tvář.

2. **Dlib HOG detektor**

   Knihovna Dlib obsahuje metodu pro detekci tváří založenou na HOG deskriptoru a SVM klasifikátoru. Pro její použití není potřeba explicitně stahovat model, protože ten je přímo zakomponován v knihovně. Podle informací uvedených v hlavičkovém souboru je model složený z pěti HOG filtrů pro tvář směřující přímo do kamery a pro mírně natočenou do různých směrů. Je natrénovaný na necelých třech tisících snímcích z LFW datové sady [58]. Metoda úspěšně detekuje tváře směřující do kamery a mírně otočené do stran. Selže však pokud je hlava otočená více do stran, nebo i jen mírně nakloněná ve vertikálním směru.

   Průměrný čas: 214 ms

   Výhody: jednoduchost modelu

   Nevýhody: nepřesná nebo špatná detekce i pod malým úhlem, vysoký čas

3. **OpenCV DNN detektor**

   Knihovna OpenCV obsahuje modul pro načtení předtrénované neuronové sítě z konfiguračního a datového souboru. Zároveň je možné stáhnout tyto soubory pro detektor tváří. Ten je založený na SSD detektoru a jako páteřní síť využívá ResNet-10 [59]. Model jako vstup potřebuje RGB obrázek s rozlišením 300 × 300 px. Metoda je schopná úspěšně detekovat také tváře, které jsou pod úhlem. Pokud je však hlava řidiče příliš předkloněná, nebo otočená do strany o více než 90°od kamery, detektor ji nenalezne. Pro použití této metody je také možné využít knihovnu Cvlib [56]. Ta obsahuje metodu, která načte zmíněný model a provede operace potřebné na předzpracování obrazu. Také následně z výstupu modelu vybere potřebná data a vrátí ohraničující rámečky a pravděpodobnosti jednotlivých detekcí. Využití této možnosti má za výhodu jednodušší kód méně náchylný k chybám, ale za cenu použití další knihovny, čímž se zvyšuje velikost aplikace. Při své implementaci jsem vyzkoušela oba zmíněné postupy použití tohoto detektoru s obdobnými výsledky.

   Průměrný čas: 34 ms

   Výhody: rychlost, detekce tváří otočených do stran i lehce nakloněných ve vertikálním směru

   Nevýhody: velikost modelu

4. **YOLO verze 3**

   Princip detektoru objektů YOLO je popsán v části 2.1.2 na straně 16. Ve své práci jsem jej implementovala s využitím OpenCV. Využila jsem funkce pro načtení Darknet modelu z konfiguračního souboru a ze souboru obsahujícího váhy, které jsem získala z [60]. Model je natrénovaný na datové sadě Wider Face [61]. Původní konfigurace modelu vyžaduje jako vstup RGB obrázek o výšce a šířce 416 px, což se při testování ukázalo jako nevyhovující. Průměrný čas potřebný na jeden snímek byl 450 ms a detekce nebyly přesné. Jako nejvhodnější rozlišení vstupu jsem testováním zjistila 220 × 220 px. V takové konfiguraci model úspěšně detekuje tváře ve většině případů. Pokud je však hlava řidiče příliš otočená do strany, nebo zakloněná detektor ji nenalezne.

   Průměrný čas: 177 ms

   Výhody: detekce tváří otočených více do stran

   Nevýhody: velikost modelu, vyšší čas, nedetekuje zakloněné tváře

Na základě výsledků z testování výše uvedených metod jsem se ve své práci rozhodla využívat OpenCV DNN detektor, protože je nejrychlejší a také nejspolehlivější pro potřeby detekování řidiče. Pokud je v obraze nalezeno více tváří, je potřeba určit, kterou z nich je řidič. Implementovala jsem jednoduchý algoritmus, který se ukázal jako dostačující. Předně se vybírají pouze detekce s pravděpodobností větší než 70 %. Vzhledem k umístění kamery a faktu, že řidič je vždy na levé straně obrazu, se vybírá detekce, jejíž ohraničující rámeček má pozici levého horního rohu

Obrázek 4.4: Fáze pohybu hlavou, ve kterých různé metody detektoru tváří selhaly. Zleva vždy: při pohybu hlavou směrem dolů, směrem do strany a zaklonění s otočením do strany. Metoda (b) u pohybu do zaklonění s otočením do strany nalezla tvář ve všech fázích.

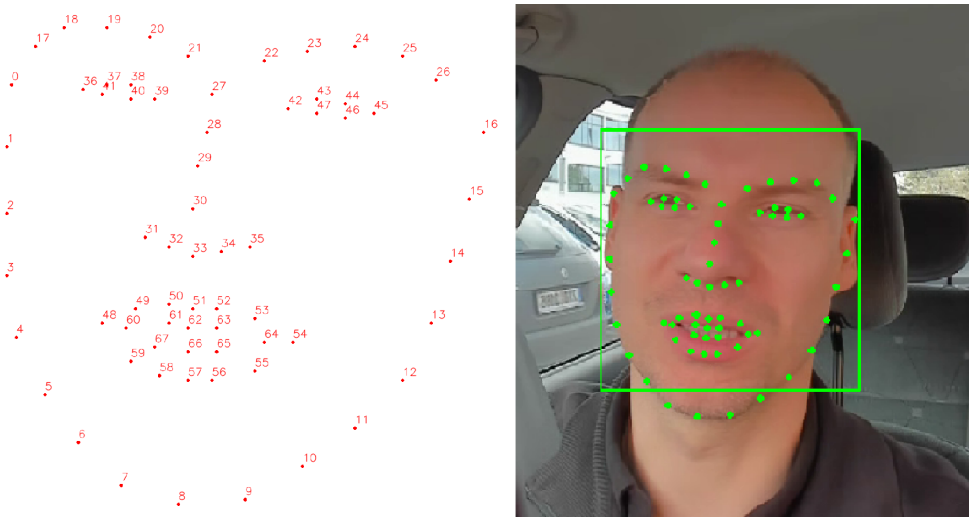
co nejvíce vlevo nahoře. Dále jsem implementovala optimalizace, které vychází z předpokladu, že hlava řidiče se v obraze za jízdy pohybuje pouze minimálně. První z nich zjednodušuje postup pro výběr správné detekce tím, že pokud je k dispozici pozice tváře v předchozím snímků, je vybrána ta, která je ji nejblíže. Dále ze vstupního obrazu používám pro detekci pouze část, ve které byl v předchozím snímku detekován řidič. Ze vstupu se ořízne oblast definovaná ohraničujícím rámečkem předchozí detekce zvětšená o definovaný počet pixelů ve všech směrech. Aby byl výsledný efekt této optimalizace co největší, snažila jsem se toto nutné zvětšení stanovit co nejmenší. Z testování vyšla jako nejvhodnější hodnota 150 px (při rozlišení vstupního obrazu 1280 $\times$ 720 px). Při nižší hodnotě již detektor selhával, nebo predikoval nepřesné rámečky. Díky tomu se snížil počet detekcí, protože ve většině případů nejsou ve výsledném obraze obsaženi ostatní lidé ve vozidle, nebo chodci za oknem. Zároveň je tvář v obraze vstupujícím do detektoru větší a tím je pro něj lépe rozpoznatelná.

4.2.2 Reprezentace dat

Pro popis detekovaného obličeje řidiče jsem se rozhodla využít orientační body obličeje. Správně predikovaná pozice těchto bodů má vysoký informační charakter a zároveň relativně malou dimenzi. To umožňuje použít menší a rychlejší model pro jejich analýzu, než by tomu bylo např. v případě zpracovávání přímo obrazových dat konvoluční sítí. Další výhodou je možnost využít natrénovaný model sloužící pro jejich analýzu i s jiným detektorem tváří, nebo prediktorem bodů. Musí být pouze zachován stejný počet a rozmístění obličejových bodů. Pokud by byl tedy implementovaný lepší algoritmus v jedné, nebo v obou těchto fázích, bylo by možné dále používat natrénovanou síť pro detekci anomálií. Nevýhodou použití orientačních bodů obličeje je však nemožnost detekovat anomálie, které nemají vliv na pozici či výraz tváře, jako je třeba telefonování. Orientační body obličeje musí být detekovány, např. využitím CNN, což do výsledné aplikace přidává další model a výpočetní krok. K tomu je možné použít některou z dostupných metod s natrénovaným modelem. Ve své práci jsem vyzkoušela dvě metody z knihoven Dlib [55] a Face Alignment [34]. Jejich princip je popsaný v části 2.2.2 na straně 21. Obě metody s využitím nabízených natrénovaných modelů predikují pozici 68 bodů, jejichž rozmístění je znázorněno na obrázku 4.5. Níže uvádím dvě testované metody, jejich výhody, nevýhody a průměrný čas na jeden snímek. Obě jako vstup dostávají celý vstupní obraz z kamery a ohraničující rámeček obličeje řidiče, který byl detekován v předchozím kroku. Výstupem je pak pole souřadnic predikovaných bodů.

1. **Dlib prediktor**

   Knihovna Dlib [55] obsahuje od verze 18.10 metodu pro predikci obličejových orientačních bodů. Jedná se o implementaci práce [32] využívající regresních stromů. Autor knihovny také nabízí datový soubor s modelem, který je natrénovaný na datové sadě orientačních bodů



Obrázek 4.5: Predikované orientační body obličeje a jejich umístění na detekované tváři.

obličeje iBUG 300-W. Metoda úspěšně a ve většině případů relativně přesně predikuje pozice bodů pro obličej, který je namířen na kameru, nebo mírně otočený. Model byl natrénovaný s detektorem tváří z knihovny Dlib, který je popsán výše. Ten, jak jsem uvedla, nefunguje pro otočené či nakloněné tváře. Proto není vhodný pro použití v této práci. V případě využití jiného detektoru, který produkuje jinak umístěné a rozdílně velké ohraničující rámečky, metoda predikuje nepřesné pozice. 2D orientační body, které tato metoda generuje, jsou navíc nepřesné tím, že kopírují pouze viditelnou část tváře. Pokud je tedy hlava otočená výrazně do strany, tak body, které by reálně měly být umístěny na odvrácené části obličeje, jsou shlukovány na viditelné linii tváře. Ilustrace je na obrázku 4.6.

Průměrný čas: 1,7 ms

Výhody: rychlost, jednoduchost modelu

Nevýhody: nepřesné predikce při použití jiného detektoru tváří než Dlib HOG, pouze 2D orientační body

2. **Face Alignment**

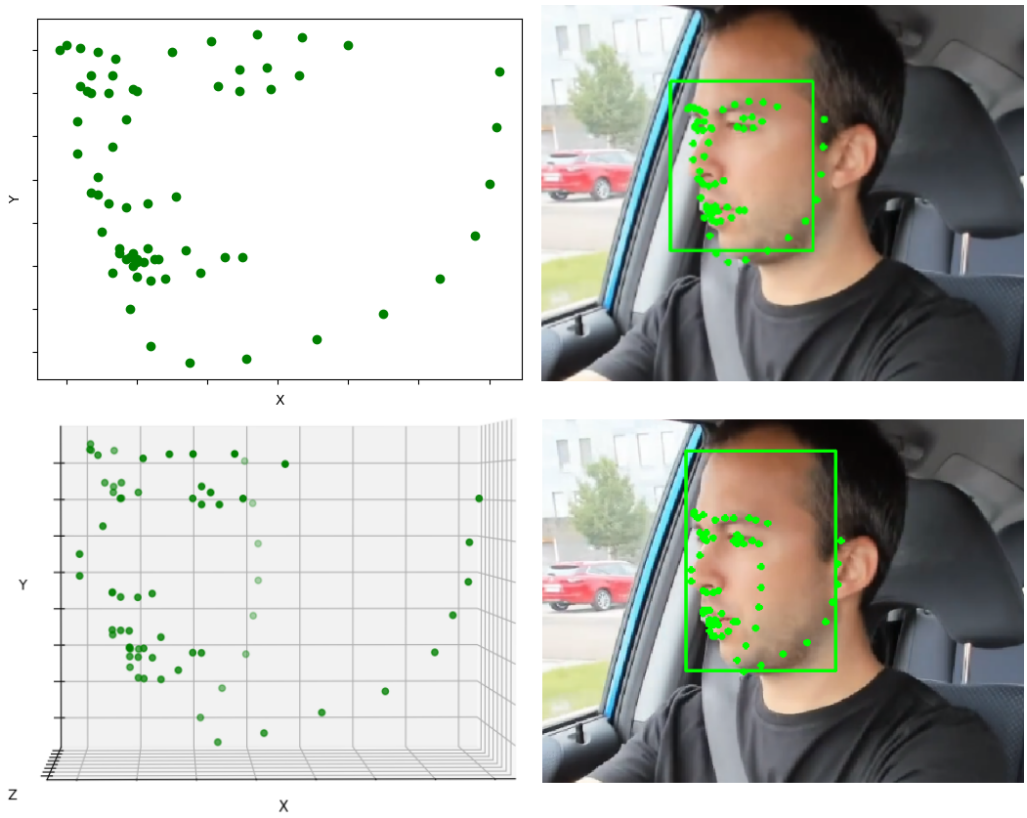
   Knihovna Face Alignment obsahuje implementace modelů a metod pro predikci orientačních bodů obličeje obsažených v práci [34]. Je možné použít model pro 2D, nebo 3D obličejové body. Výhodou 2D prediktoru je nižší čas, ale 3D body lépe popisují reálnou pozici hlavy. Ilustrace je na obrázku 4.6. Přínos takovýchto 3D orientačních bodů je znatelný jen v případě otočené hlavy do strany. Pokud je obličej namířen směrem do kamery tak, že jsou všechny predikované body umístěny na viditelné části tváře jsou 2D a 3D orientační body shodné. Řidič vozidla ale hlavou často otáčí a je snímán kamerou mírně ze strany. Proto má použití 3D typu predikovaných bodů v navrhované aplikaci význam. Výhodou metody je možnost použití různých detektorů tváře aniž by to ovlivnilo její výkon. Testováním jsem zjistila, že nejčastějším problémem je nepřesná predikce pozice bodů definujících čelist a obočí. Příklady takových chybných predikcí jsou na obrázku 4.7.

   Průměrný čas: 3D - 136 ms, 2D - 84 ms

   Výhody: 3D orientační body, dobré predikce při použití různých typů detektorů tváří

   Nevýhody: vyšší čas, velikost modelu

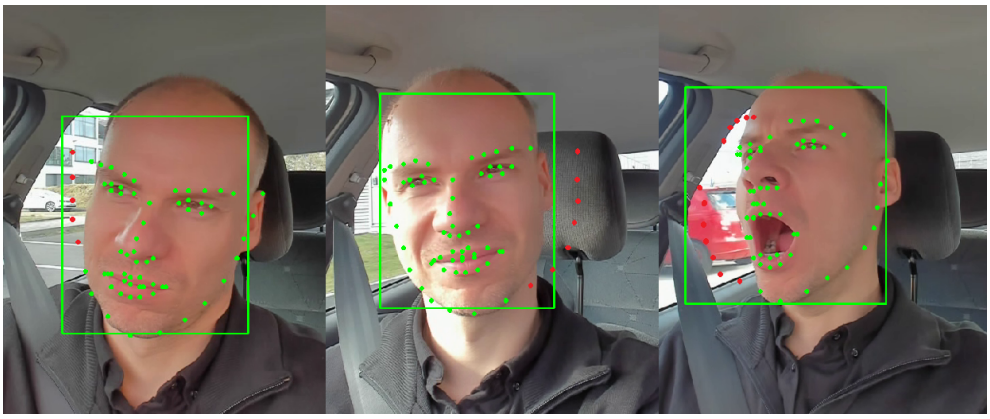
Z těchto dvou testovaných metod jsem se rozhodla využívat 3D obličejové body predikované metodou Face Alignment, protože pro fungování aplikace je nezbytné, aby byly body správně detekované i v případě otočené či nakloněné tváře. Využívám pouze jejich 2D projekci, protože pro popis sledované tváře je to dostačující. Při analýze pozice a výrazu tváře je irelevantní jeho vzdálenost od kamery, tedy velikost. Za tímto účelem jsem implementovala normalizaci tak, aby výsledné orientační body obličeje měly x a y souřadnice v rozmezí $\langle 0, 1 \rangle$.



Obrázek 4.6: Porovnání 2D (nahoře) a 3D (dole) orientačních bodů obličeje.

Pro samotnou analýzu následně vybírám pouze určitou podmnožinu ze všech 68 bodů tak, aby dostatečně popisovaly sledovanou tvář a zároveň zbytečně nezvětšovaly potřebnou velikost modelu pro jejich analýzu. Při rozhodování jsem se soustředila na to, co má být z výsledné reprezentace rozeznatelné a také na to, abych se pokud možno vyhnula bodům u nichž prediktor často chybuje. Prvně jsem se tedy rozhodla nepoužívat body popisující obočí a čelisti, protože u těchto dochází ke značně nepřesným predikcím, jak jsem uvedla výše. Pro určení pózy tváře a jejich částí jsou nejdůležitější oči a ústa. Vzhledem k tomu, že každý člověk má oči jinak velké či přivřené není vhodné používat všechny body, které je popisují. Metoda navíc není u těchto bodů dostatečně přesná na to, aby bylo možné detekovat např. zavřené oči. Z toho důvodu pro popis očí používám pouze jejich vypočtený střed. První reprezentace, kterou jsem zkoušela, byla pomocí tří bodů: střed levého oka, střed pravého oka a střed úst. To se ukázalo jako dobrá reprezentace pro popis pózy obličeje, ze které šlo určit jeho naklonění či otočení do strany. Nebylo však možné určit otevření úst např. při zívání, nebo dušení řidiče. Proto jsem se pro popis úst rozhodla použít čtyři body umístěné na vnějších stranách. Do reprezentace jsem také přidala bod popisující špičku nosu, která reprezentuje střed obličeje a pomáhá lépe odrážet pozici tváře. Výsledná reprezentace pózy obličeje tedy sestává ze 7 bodů o souřadnicích x a y.

Ze všech trénovacích videí jsem extrahovala normalizované orientační body obličeje z detekované



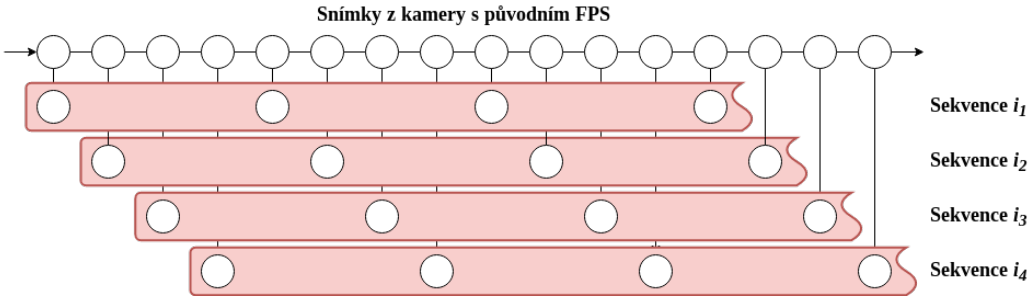
Obrázek 4.7: Příklady nepřesných predikcí orientačních bodů obličeje metodou Face Alignment. Chybné body jsou označeny červenou barvou.

tváře řidiče. Ty jsem pro každé video uložila do samostatného souboru. Pokud pro určitý snímek videa nebyl detekován obličej řidiče, do souboru se uložily hodnoty NaN (Not a Number). Takových snímků bylo ve všech videích celkem 3585, což je 1,25 %. Vzhledem k tomu, že detekce orientačních bodů je časově nejnákladnější operací celé navrhované aplikace, předzpracováním videí se značně zjednodušilo trénování sítě.

4.2.3 Analýza dat

Pro účely analýzy dat jsem testovala dvě metody využívající hluboké neuronové sítě. K implementaci modelů jsem využila API Keras [62] a Tensorflow backend [63]. První z nich je variační autoenkodér složený z LSTM vrstev, který má za úkol se naučit co nejlépe rekonstruovat normální sekvence. Druhou je RNN model pro predikci blízké budoucnosti na základě sekvence na vstupu. Obě metody jsou založené na principu trénování pouze na normálních datech tak, aby pro anomální vstup produkovaly vysokou chybu. Vstupem pro obě metody jsou sekvence vybraných obličejových bodů o definované délce. V této fázi bylo potřeba určit požadované FPS, aby byla trénovací data správně navzorkovaná. Vzhledem k průměrnému času potřebnému na získání obličejových bodů z jednoho snímku, který je 170 ms, jsem zvolila FPS 5, abych umožnila běh aplikace v reálném čase (na GPU). Aby bylo pro trénování dostatečné množství dat, využívám všechny snímky původních videí, které mají FPS 30. Algoritmus pro vzorkování nejprve navzorkuje data na požadované FPS postupem, který je znázorněný na obrázku 4.8. Tímto vznikne z původní jedné sekvence s FPS 30, šest navzorkovaných s požadovaným FPS 5. Z těch se metodou posuvného okna s posunem jedna vytvoří sekvence, které slouží jako vstup do modelu. Aby byla výsledná data smysluplná musí být zachován sled snímků z kamery. Pokud je tedy v souboru s obličejovými body nalezen snímek, který má NaN hodnoty, nemůže se pouze přeskočit, protože by se narušila posloupnost dat. V takovém případě se proto ukončí vzorkování a z prozatím získaných dat se vytvoří finální sekvence. Následně se pokračuje ve vzorkování dalších dat od následujícího snímku, ve kterém již byly body prediko-

vané. Takto vytvořené sekvence ze všech trénovacích videí, které představují finální datovou sadu, jsou opět uložené do souboru. Pro RNN síť se navíc ještě vytváří soubor obsahující pro každou sekvenci snímek v definovaně vzdálené budoucnosti, na základě kterých se síť učí predikovat.



Obrázek 4.8: Ilustrace postupu vzorkování trénovacích dat.

Implementované modely jsem testovala na testovací datové sadě. Výstupem je pro každé video soubor obsahující anomální skóre pro jednotlivé snímky při FPS 5. Pokud při testování videa pro aktuální snímek chybí orientační body obličeje (při výpadku detektoru tváří) je jako anomální skóre použito to, které bylo predikované pro předchozí snímek. Takto jsem to implementovala z důvodu, že detektor tváří selhává jak pro normální data, např. z důvodu špatného osvětlení, tak pro anomální data, např. při značně předkloněné hlavě při spánku. Díky tomu jsou snímky, při kterých dojde k výpadku detektoru způsobenému anomálií označeny za anomální. Testované metody jsem hodnotila pomocí několika metrik. Nejprve jsem na základě predikovaných a skutečných anomálií získala počty snímků správně pozitivních (TP), správně negativních (TN), falešně pozitivních (FP) a falešně negativních (FN). Přičemž pozitivní zde znamená anomální. Jako první metriku jsem zvolila nejčastěji používanou přesnost (Accuracy ACC), která je definována následovně:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{4.1}$$

ACC tedy vyjadřuje poměr úspěšných detekcí ku všem ohodnoceným snímkům. Jako další metriku používám F1 skóre, které je definováno jako:

$$\mathrm{F1} = 2 \times \frac{\mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}, \tag{4.2}$$

přičemž Recall a Precision jsou metriky definované jako:

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{4.3}$$

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{4.4}$$

F1 skóre bere v potaz také falešně pozitivní a negativní detekce a je tedy lepším ukazatelem spoleh-

livosti metody. Pro hodnocení schopnosti metody detekovat jednotlivé třídy anomálií jsem použila Recall RCL_x, který se počítá podle vzorce výše s hodnotami týkajícími se pouze dané třídy x.

1. **VAE**

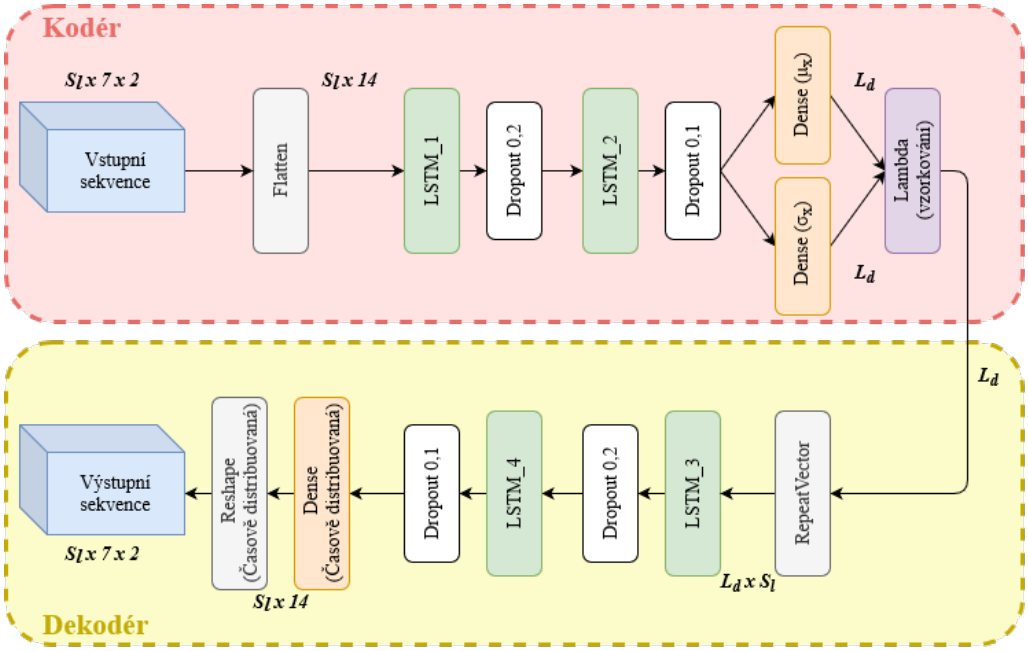
   Variační autoenkodér jsem implementovala za účelem zpracování a rekonstrukce sekvencí dat. Z toho důvodu jsem jako vrstvy kodéru a dekodéru použila LSTM. Vstupem do kodéru jsou sekvence (časové řady) 7 obličejových bodů se souřadnicemi x a y. Jako nejvhodnější počet snímků v sekvenci S_l jsem testováním zjistila hodnotu 5. Při FPS 5 je to tedy jedna sekunda videa. První vrstvou základního modelu je Flatten vrstva, která z pole 2D bodů vytvoří 1D pole, které je seřazením původních souřadnic jednotlivých bodů. Následuje LSTM vrstva, která má za vstup sekvenci takto zploštěných dat o rozměrech $S_l \times 14$. Na ní jsou napojeny dvě Dense vrstvy o velikosti latentního prostotu, jejichž výstup představuje střední hodnotu μ_x a směrodatnou odchylku σ_x rozložení, do kterého byla vstupní sekvence zakódována. Z těchto dvou vrstev se vzorkují data z definovaného rozložení podle rovnice (2.6). Následuje vrstva RepeatVector, která z definovaného latentního prostoru pomocí vzorkovací funkce získá data k dekódování sekvence. Poté už následuje dekodér, který je zrcadlením modelu kodéru, tedy LSTM vrstva, Dense vrstva s velikostí zploštěného vstupu a nakonec Reshape vrstva, která výstup dekodéru převede zpět do dimenze vstupních dat. Dense a Reshape vrstvy jsou časově distribuované, což umožňuje zpracování sekvencí. Jako chybu modelu, na základě které se síť učí, jsem použila součet KL divergence podle rovnice (2.7) a střední kvadratické odchylky (MSE) mezi vstupní a výstupní sekvencí, která je definována jako:

$$\text{MSE} = \frac{1}{MN} \sum_{M,N} (x_{m,n} - \overline{x}_{m,n})^2 + (y_{m,n} - \overline{y}_{m,n})^2, \tag{4.5}$$

   kde M v mém případě představuje počet snímků v sekvenci, N počet bodů v jednom snímku, x a y jsou souřadnice těchto bodů ve vstupní sekvenci a $\overline{x}$ a $\overline{y}$ ve výstupní.

   Celkový počet sekvencí pro trénování při $S_l = 5$ byl necelých 273 tisíc. Z toho 10 % jsem používala pro validaci v průběhu trénování. Použila jsem optimalizér RMSprop, který je implementovaný v Kerasu, s poměrem učení $1e^{-4}$. Vzhledem k objemu dat jsem velikost dávky (batch size) nastavila na 128. Počet epoch jsem nadefinovala na 100 a abych předešla přeučení sítě, použila jsem tzv. early stopper, který trénování zastaví, pokud se validační chyba nezlepší alespoň o jednu desetitisícinu za uplynulých 10 epoch. Body ve vstupních datech byly v rozmezí $\langle 0, 1 \rangle$, za aktivační funkci jednotlivých vrstev jsem proto volila ReLu. Takto nadefinovaný model se zpočátku učil velmi rychle (trénovací i validační chyba rychle klesala). Po pár epochách však byla KL divergence skoro nulová, rekonstrukční chyba přestala klesat a trénování bylo ukončeno. Z vizualizace výstupních dat modelu se ukázalo, že síť se naučila rekonstruovat všechny vstupy jako sekvence jen minimálně se měnících dat, které by se

daly označit za průměrnou pozici obličejových bodů v trénovacích datech. Zkusila jsem proto implementovat hlubší model přidáním druhé LSTM vrstvy (o stejné velikosti jako první) do dekodéru i kodéru a Dropout vrstvy o poměru výpustě $0,2$ mezi LSTM vrstvy a $0,1$ za poslední LSTM. Ilustrace takového modelu je na obrázku 4.9. Ani tato architektura modelu však nebyla úspěšná. Síť jsem také nechala trénovat po celých 100 epoch, ale ani po této době se rekonstrukce nezlepšovala. Z vizualizace výstupů jednotlivých vrstev bylo patrné, že kodér v datech nenachází žádný vzor a výstup z latentního prostoru je nezávisle na vstupní sekvenci stále stejný.



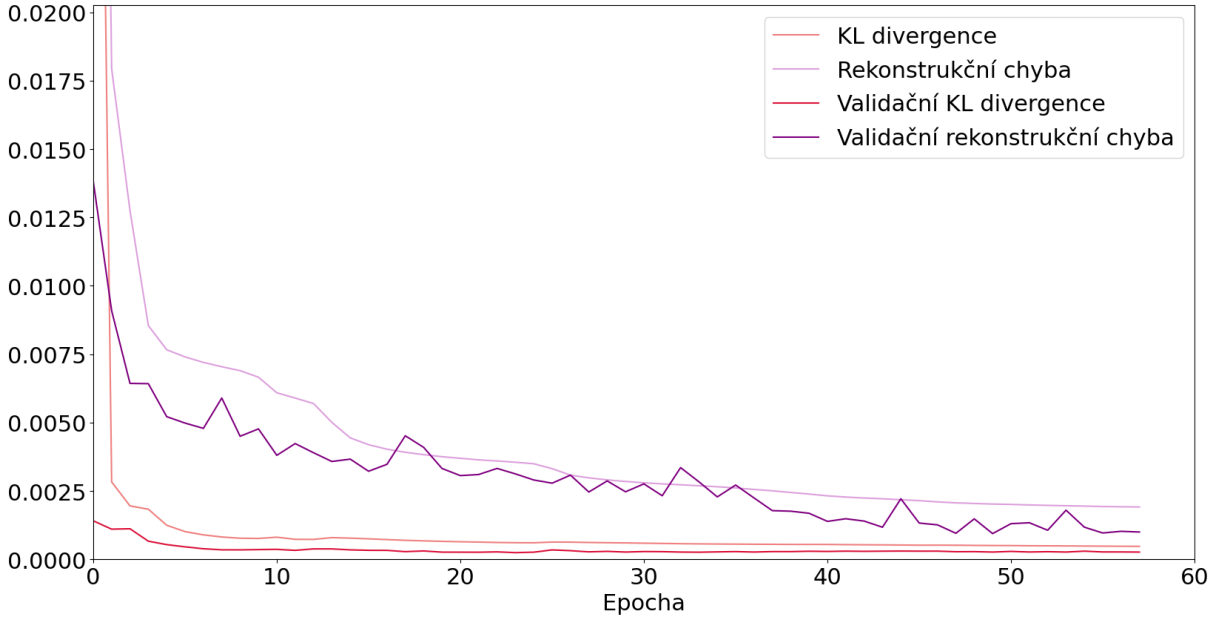
Obrázek 4.9: Ilustrace testované architektury VAE, kde S_l je délka vstupní sekvence a L_d je velikost latentního prostoru.

Vstupní data jsem změnila tak, aby souřadnice byly v rozmezí $\langle -1, 1 \rangle$. Spolu s tím jsem v modelu změnila aktivační funkci jednotlivých vrstev na hyperbolický tangent, který je definován jako:

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{4.6}$$

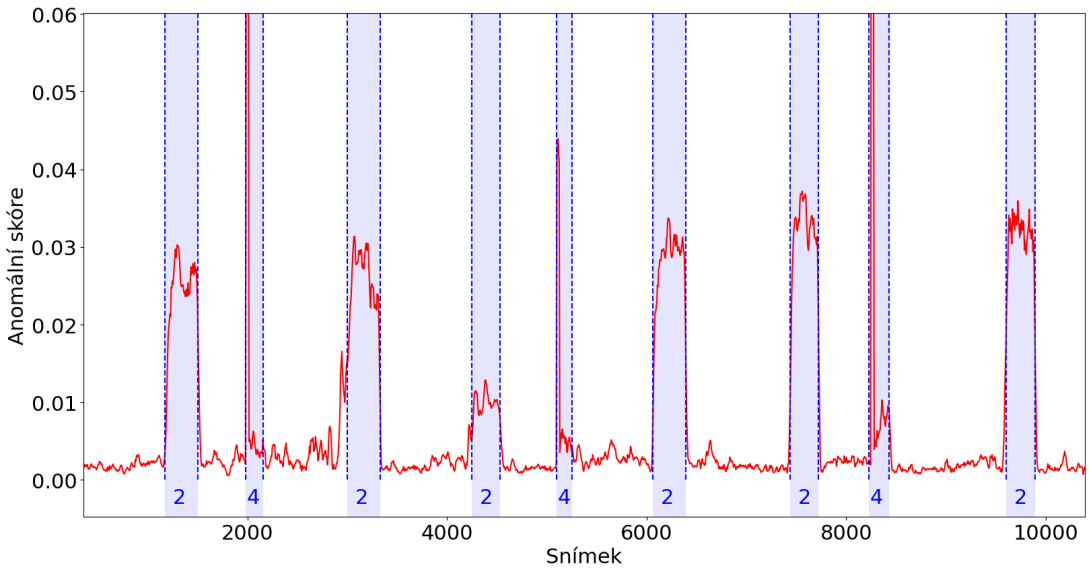
Po této úpravě se již navržený model začal správně učit kódovat i rekonstruovat vstupní data. Architektura modelu pouze s jednou LSTM sítí dosahovala horších výsledků, proto se zde dále budu zabývat pouze modelem s dvěma vrstvami a výpustěmi tak, jak je popsán výše a znázorněn na obrázku 4.9. Testováním jsem zjistila jako nejlepší velikost latentního prostoru 32. Za velikosti jednotlivých rekurentních vrstev modelu jsem testovala více hodnot. Nejlepších výsledků dosahovaly modely, které měly všechny vrstvy stejně velké v rozsahu od 64 do 256.

Příklad vývoje chyby v průběhu trénování jednoho z modelů je zachycen v grafu na obrázku 4.10. Je z něj patrná poměrně rychlá stabilizace KL divergence, která se mírně zvětšuje vždy, když rekonstrukční chyba výrazněji klesá.



Obrázek 4.10: Graf zachycující chybu modelu VAE v průběhu trénování.

Testování modelů probíhalo na testovacích datech, kde bylo celkem 14 619 ohodnocených snímků. Jako anomální skóre jsem použila chybu modelu. Příklad grafu zachycujícího anomální skóre jednoho testovacího videa je na obrázku 4.11. Na obrázku 4.15 jsou pak zachyceny ROC křivky několika testovaných modelů. Z křivek týkajících se variačních autoenkodérů je patrné, že nejlepších výsledků při míře falešně pozitivních detekcí menší něž 0,18 dosahoval model s velikostí LSTM vrstev 128. Při vyšší míře falešně pozitivních detekcí byl nejúspěšnější největší model s velikostí LSTM vrstev 256. Pro hodnocení modelů jsem dále počítala zvolené metriky. Bylo potřeba zvolit mezní hodnotu, která určuje při jaké velikosti anomálního skóre se již jedná o anomálii. Pro její určení jsem využila data z ROC křivky a použila takovou mezní hodnotu, při které byla míra falešně pozitivních detekcí 10 %. Podrobné výsledky modelů při takto nastavých mezních hodnotách jsou v tabulce 4.2. Je z ní patrné, že nejhůře bylo detekováno telefonování, protože to z obličejových orientačních bodů nelze poznat. U takových anomálií je detekce úspěšná pouze v případech, kdy je způsoben anomální pohyb, nebo pozice hlavy, případně špatná predikce orientačních bodů obličeje, která taktéž způsobí vysoké anomální skóre.



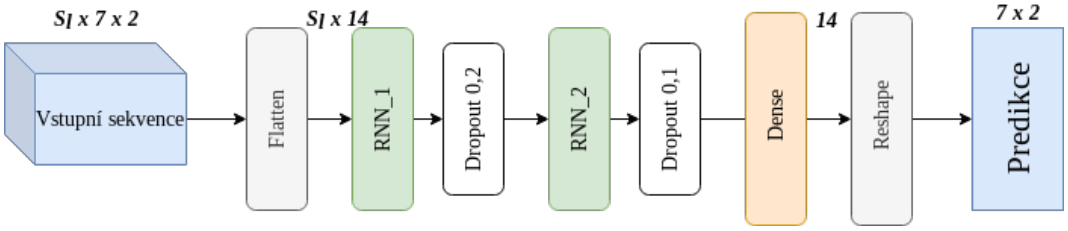
Obrázek 4.11: Ukázka grafu zachycujícího anomální skóre testovacího videa při použití VAE. Modře vyznačené oblasti jsou skutečné anomálie videa, číslo představuje příslušnou třídu.

2. **Predikční RNN**

Rekurentní neuronovou síť jsem implementovala za účelem predikce budoucího snímku na základě vstupní sekvence. Vstupem do modelu je tedy sekvence orientačních bodů obličeje vytvořená z několika po sobě jdoucích snímků. Jako délku této sekvence jsem testováním zvolila stejně jako u VAE hodnotu $S_l = 5$. Vyšší hodnoty zvětšovaly model bez pozitivního přínosu na výsledek. Nižší hodnoty zase nenesly dostatečnou informaci o prováděné akci, a tak byly výsledky opět horší. Při zvolené hodnotě $S_l = 5$ se tedy zpracovává jedna sekunda videa, což je dostatečná doba na popis prováděné akce. Dalším parametrem, který bylo potřeba zvolit, je vzdálenost predikované budoucnosti P_t. Tedy kolikátý snímek následující za posledním obsaženým v analyzované sekvenci se predikuje. Opět jsem testovala několik hodnot. Nejlepších výsledků dosáhly modely s parametrem $P_t = 2$. Při vyšších hodnotách se modely trénovaly špatně, nejspíše proto, že vzhledem ke snímkové frekvenci zpracovávaného videa už očekávaná predikce dostatečně nevycházela ze vstupní sekvence. Při $P_t = 1$ byly výsledky modelů o něco málo horší. Při takto nastavených parametrech bylo trénovacích sekvencí bez mála 270 tisíc. Z toho 10 % jsem použila k validaci v průběhu trénování. Stejně jako v případě VAE jsem použila optimalizér RMSprop s poměrem učení $1e^{-4}$, velikost dávky 128 a tzv. early stopper, který trénování zastaví, pokud se validační chyba nezlepší alespoň o jednu desetitisícinu za uplynulých 10 epoch.

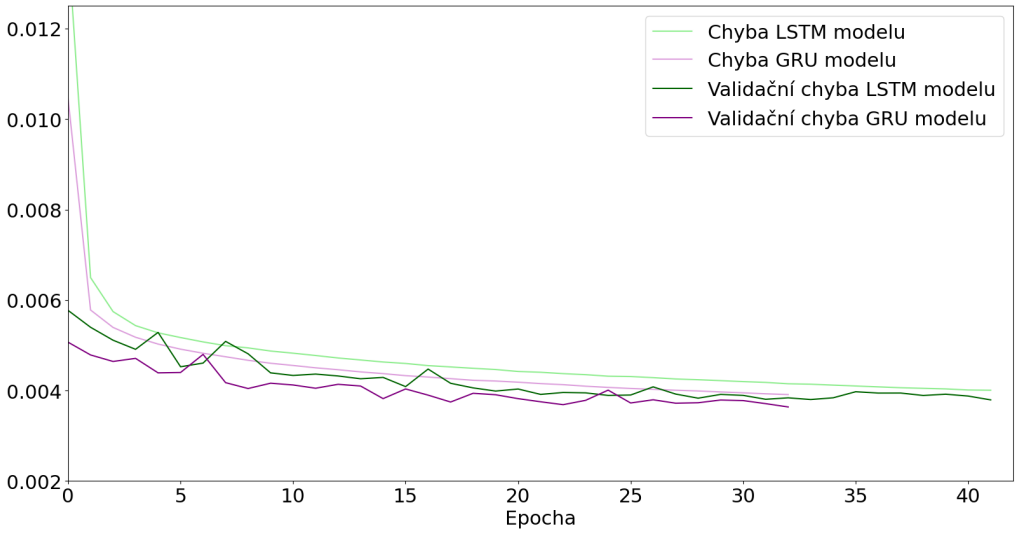
Na začátku modelu sítě je stejně jako u VAE Flatten vrstva, která ze vstupního pole 2D bodů vytvoří 1D pole. Následuje RNN vrstva, která je složená z LSTM, nebo GRU buněk. Za ní je zařazena Dense vrstva, jejímž výstupem je predikce o dimenzi $S_l \times 14$. Tento výsledek je

pomocí Reshape vrstvy přeměněný do očekávané dimenze 2D bodů, tedy $S_l \times 7 \times 2$. Jako chybu modelu jsem zvolila střední kvadratickou odchylku mezi očekávanými a predikovanými orientačními body. Tento základní model se neučil dostatečně dobře nehledě na velikost vrstev. RNN vrstvy jsem proto použila dvě s Droupout vrstvou s poměrem výpustě $0,2$ mezi nimi a $0,1$ za poslední. Ilustrace architektury modelu je na obrázku 4.12.



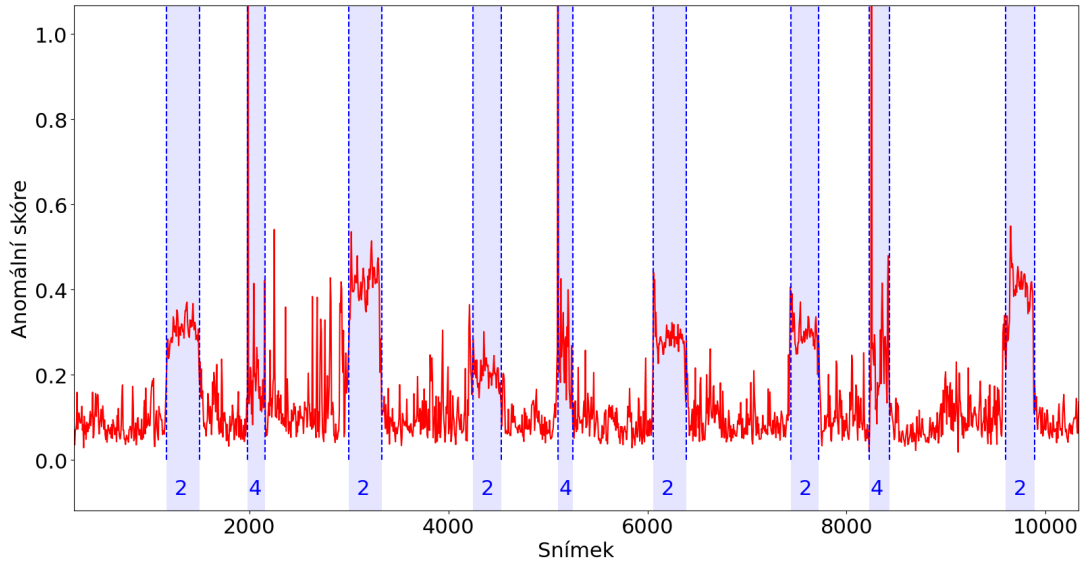
Obrázek 4.12: Ilustrace testované architektury RNN modelu, kde S_l je délka vstupní sekvence.

Testovala jsem modely složené z LSTM a GRU buněk. Oba typy sítí dosáhly obdobných výsledků, ale lišily se velikostí (počtem parametrů). Pro příklad síť složená ze dvou LSTM vrstev s velikostí 256 obsahuje přes 800 tisíc trénovatelných parametrů. Stejná síť složená z GRU vrstev má 605 tisíc parametrů, tedy o skoro 200 tisíc méně. Trénování po jednu epochu trvalo průměrně 60 sekund u obou typů sítí. LSTM model se trénoval po 42 epoch, zatímco GRU pouze po 32. GRU síť měla navíc v průběhu celého trénování nižší chybu, jak je patrné z grafu na obrázku 4.13, který zachycuje chybu těchto modelů v průběhu trénování. Protože jediný výraznější rozdíl mezi použitím LSTM a GRU vrstev je velikost modelu a s tím spojená potřebná doba na trénování, je výhodnější používání modelu složeného z GRU vrstev.



Obrázek 4.13: Graf zachycující chybu GRU a LSTM modelů se stejnou architekturou v průběhu trénování. Je vidět, že LSTM model měl o něco větší chybu v celém průběhu a trénoval se déle.

Při testování modelu výsledná chyba velmi kolísala a nebylo možné určit mezní hodnotu pro anomálie. Z vizualizace jsem zjistila, že model při rychlejších pohybech hlavou do strany neuměl správně určit souřadnice x v predikované budoucnosti. Pozice byly zjednodušeně řečeno zpožděné oproti skutečným, což způsobovalo vysokou chybu i u normálních dat. Model se tedy nedokázal dostatečně naučit promítnout do výsledných dat rychlost pohybu ze vstupní sekvence. To přikládám faktu, že v trénovacích datech je výrazně více statických sekvencí, kdy se řidič dívá před sebe a pohybuje při tom hlavou pouze minimálně. Tento problém nastal při $P_t = 2$ a také při $P_t = 1$. Model, který by eliminoval tento problém se mi nepovedlo implementovat. Ukázalo se však, že pro posouzení toho, zda se jedná o anomálii či ne, je důležitá póza hlavy a ne její aktuální otočení do strany. Implementovala jsem proto výpočet anomálního skóre jako součet odchylek bodů mezi skutečnými a predikovanými pouze ve směru osy y. S takto definovaným skóre již bylo možné určit mezní hodnotu a detekovat anomálie. Tento postup je poměrně dobře aplikovatelný na použitá testovací data. Není ovšem dostatečně obecný a nové neznámé anomálie by nemusely být detekovány.



Obrázek 4.14: Ukázka grafu zachycujícího anomální skóre testovacího videa při použití predikční GRU sítě. Modře vyznačené oblasti jsou skutečné anomálie videa, číslo představuje příslušnou třídu.

Na obrázku 4.15 jsou zachyceny ROC křivky úspěšných testovaných modelů. Nejlepších výsledků při míře falešně pozitivních detekcí menší než 0,14 dosahoval model s GRU vrstvami s velikostí 64. Větší modely dosahovaly lepších výsledků při vyšší míře falešně pozitivních detekcí. Při následném výpočtu zvolených metrik jsem volila mezní hodnotu na základě dat z ROC křivky tak, aby míra falešně pozitivních detekcí byla 10 %. Nejlepších výsledků dosáhl GRU model s velikostí obou vrstev 64. Ten byl zároveň nejúspěšnější v detekci většiny tříd. Podrobné výsledky modelů s oběma typy vrstev s různými velikostmi jsou v tabulce 4.2. Je

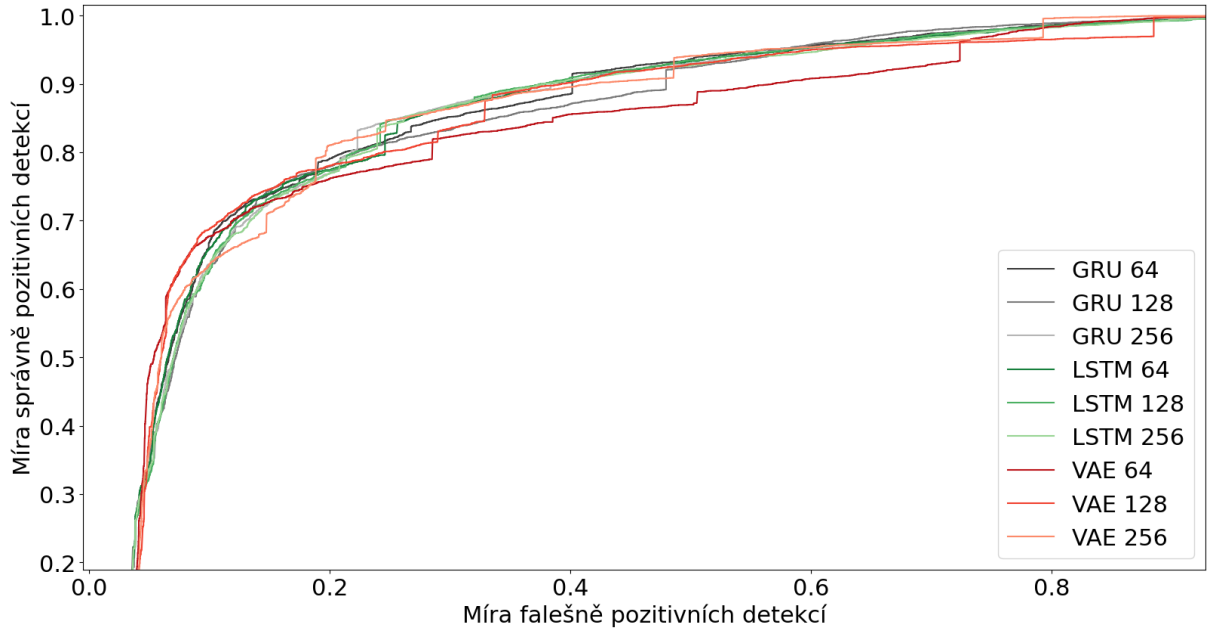
z ní patrné, že výsledky různých modelů se liší pouze minimálně. Nejhůře detekovaná třída je stejně jako u variačního autoenkodéru telefonování, což je vzhledem k použití orientačních bodů obličeje očekávané. Anomální skóre pro normální části testovacích videí bylo velmi kolísavé, jak je vidět na obrázku 4.14, který zachycuje anomální skóre jednoho testovacího videa. I přes to, že modely dosahují slušných výsledků při správně nastavené mezní hodnotě pro testovací data, dá se očekávat, že pro nová videa by byla detekce při použití stejné mezní hodnoty značně nepřesná.

Výhodou použití RNN modelu pro predikci je ta, že při krátkodobém výpadku detektoru tváře může být místo skutečných hodnot použitá predikce aktuálního snímku. Predikce se v takovém případě uloží do sekvence a evaluace vstupu může pro další snímky pokračovat. Predikci ale nelze použít pokud se jedná o anomálii, protože z podstaty modelu je v takovém případě špatná. Pokud je tedy snímek předcházející tomu, který chceme predikovat označený za anomální, nelze ji použít. Dále nelze používat predikce při delším výpadku detektoru, protože by už výsledné sekvence mohly být příliš vzdálené skutečnosti a analýza by neměla význam. Za maximální počet použitých predikcí v jedné sekvenci jsem zvolila tři.

Tabulka 4.2: Výsledky testování modelů pro detekci anomálií. Velikost RNN značí velikost rekurentních vrstev modelu. Použité metriky jsou popsány v úvodu části 4.2.3, jejich hodnoty jsou uváděné v procentech. Význam tříd anomálií (1-5) je v tabulce 4.1. Mezní hodnoty pro určení anomálie byly nastaveny tak, aby míra falešně pozitivních detekcí byla 10 %.

Model	Velikost RNN	ACC	F1	RCL_1	RCL_2	RCL_3	RCL_4	RCL_5
LSTM VAE	64, 64	83,08	71,25	81,53	73,54	86,19	51,45	27,02
	128, 128	83,41	71,95	81,02	66,94	93,09	72,06	22,21
	256, 256	81,81	68,40	58,47	66,03	96,70	85,52	19,80
GRU	64, 64	82,79	70,63	91,39	58,26	85,14	53,15	27,40
	128, 128	81,72	68,21	91,02	51,50	75,68	51,11	27,97
	256, 256	81,84	68,47	90,66	50,67	81,68	49,91	27,54
LSTM	64, 64	82,51	70,01	91,61	54,17	84,83	54,17	27,54
	128, 128	81,49	67,68	90,95	49,92	80,63	49,23	22,88
	256, 256	81,62	67,97	91,75	49,58	81,68	49,74	23,02

Z uvedených metod jsem se rozhodla využívat variační autoenkodér. Predikční rekurentní sítě sice dosáhly srovnatelných výsledků, ale jak jsem již uvedla, anomální skóre se počítá jako odchylka bodů pouze ve směru osy y, což není dostatečně obecné. U jiných testovacích dat při použití stejné mezní hodnoty by se proto dalo předpokládat, že výsledky budou horší. Z testovaných modelů VAE jsem se rozhodla používat ten s velikostí rekurentních vrstev 128, protože dosáhl nejlepších výsledků se zohledněním potřeby minimalizovat falešně pozitivní detekce.

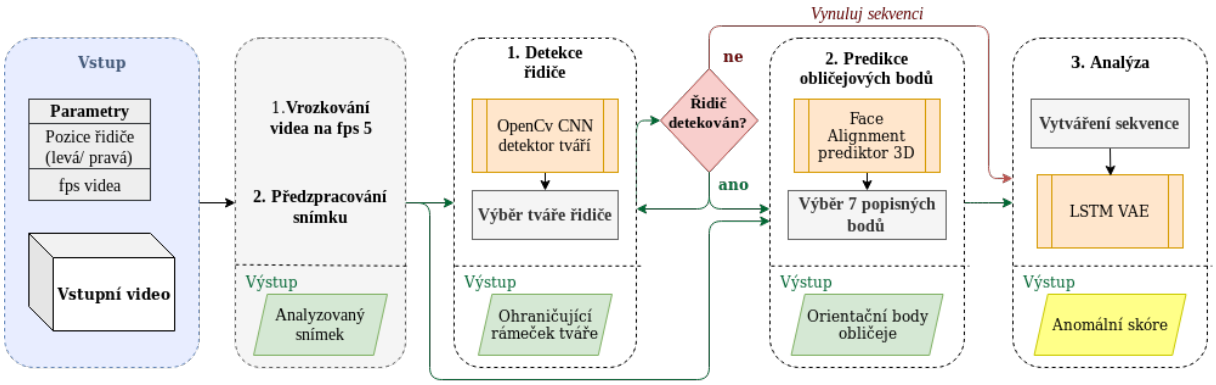


Obrázek 4.15: ROC křivky testovaných modelů pro detekci anomálií (přiblížený levý horní roh).

4.3 Navržená aplikace

Níže v této části uvádím pouze shrnutí fungování výsledné aplikace a dosažených výsledků. Detailní popis metod použitých v jednotlivých částech navržené aplikace se zdůvodněním jejich výběru a nastavení parametrů je popsáno výše v části 4.2.

Výsledná aplikace pro detekci anomálních stavů řidiče vozidla zpracovává RGB video, které zachycuje řidiče při řízení vozidla. Kamera produkující vstupní video je umístěná na vnitřním zpětném zrcátku. Pro úspěšné fungování by měla řidiče zachytávat z podobného úhlu jako ve videích použitých při vytváření aplikace a trénování modelů. Video se zpracovává při šířce 1280 px. Každý vstupní snímek je na začátku zpracování změněný na tuto definovanou velikost. Rozlišení vstupního videa není přesně definováno, ale v případě příliš malého rozlišení by mohlo docházet k nepřesnostem v částech detekce obličeje a obličejových bodů. Dále musí být aplikaci poskytnuta informace, zda je řidič ve vozidle na levé či pravé straně z pohledu kamery. Pokud se nachází na pravé straně, tak se vstupní snímky vodorovně překlápí. Dalším potřebným parametrem je FPS vstupního videa, které musí být alespoň pět, což je snímková frekvence, při které aplikace data analyzuje. Při FPS větším než nutných 5 se vstupní data vzorkují. Předzpracované snímky z videa jsou postupně vstupem do samotného detektoru anomálií, jehož postup je popsán níže a znázorněn na obrázku 4.16.



Obrázek 4.16: Diagram postupu navrženého detektoru anomálií.

- **Postup detektoru anomálií**

1. Prvním krokem aplikace pro každý analyzovaný snímek videa je detekce tváře řidiče. To probíhá pomocí detektoru tváří z knihovny OpenCV [57], který využívá natrénovaný CNN model. Z výsledných detekcí se vybere ta, která představuje řidiče. Pokud je dostupný ohraničující rámeček detekovaný v předchozím snímku, tak se využívá pro jednodušší detekci i následný výběr správné detekce. Pokud není ve snímku tvář řidiče detekovaná, aplikace nemůže dále pokračovat a přechází na další snímek. Tato část je tedy kritická pro fungování celé aplikace. Průměrný čas potřebný na detekci a výběr tváře z jednoho snímku je 34 ms. Výstupem je pozice a velikost ohraničujícího rámečku v analyzovaném snímku.
2. Dalším krokem aplikace je predikce pozice orientačních bodů obličeje. K tomu se využívá natrénovaný model pro predikci 3D orientačních bodů, který je dostupný v knihovně Face Alignment [34]. Vstupem do metody je analyzovaný snímek a ohraničující rámeček detekovaný v předchozím kroku. Výstupem je pak 3D pozice 68 obličejových bodů, přičemž pro další zpracování se používá pouze jejich 2D projekce. Následně se body normalizují tak, aby byly hodnoty souřadnic v rozmezí $\langle -1, 1 \rangle$. Nakonec se vytvoří soubor sedmi bodů, které tvoří středy levého a pravého oka, špička nosu a čtyři body popisující ústa. Takto vytvořené pole je výstupem této části aplikace a slouží jako popis řidiče v analyzovaném snímku. Nepřesná predikce těchto bodů má tedy nevyhnutelně za důsledek nepřesnou analýzu. Průměrný čas potřebný na zpracování jednoho snímku je 130 ms.
3. Posledním krokem aplikace je samotná analýza. Nejprve se z obličejových orientačních bodů generovaných v předchozím kroku vytváří sekvence sestávající z pěti po sobě jdoucích snímků, přičemž poslední je aktuálně analyzovaný snímek. Pokud pro aktuální snímek nejsou body dostupné v důsledku výpadku detektoru tváří, začíná vytváření sekvence od začátku a analýza tak nemůže probíhat pro aktuální ani pro čtyři následující snímky.

V takovém případě je anomální skóre aktuálního snímku rovno anomálnímu skóre snímku předchozího. Úspěšně vytvořená sekvence se analyzuje pomocí natrénovaného variačního autoenkodéru složeného z LSTM vrstev o velikosti 128. Ten byl natrénovaný pouze na normálních sekvencích. Výsledné anomální skóre analyzovaného snímku tvoří výsledná chyba VAE, kterou tvoří rekonstrukční chyba (MSE) a KL divergence. Mezní hodnota, která určuje zda se již jedná o anomálii, je nastavená na hodnotu 0,0031. Potřebný čas na analýzu jedné sekvence je v jednotkách milisekund.

- **Výsledky aplikace na testovacích datech**

  Pro soubor testovacích videí je výstupem aplikace celkem 14 619 snímků ohodnocených anomálním skóre. Četnost a typy anomálií v testovacích videích jsou v tabulce 4.1. Anomálních snímků je v datech celkem 4 528, normálních pak 10 091. Procento správných detekcí aplikace je následující (pozitivní zde znamená anomální):

  – Správně pozitivní detekce: 68,73 % ze všech anomálních snímků
  – Správně negativní detekce: 89,99 % ze všech normálních snímků

  Nejlépe ze všech anomálií aplikace detekuje zívání, pro které má úspěšnost detekce 93,09 %. To je způsobeno otevřenými ústy, které se v normálních datech neobjevují a proto je VAE rekonstruuje jako zavřená, což způsobí velkou odchylku u všech čtyř bodů, které je popisují. Často je navíc při zívání také způsobena anomální póza hlavy. Další dobře detekovanou anomálií je předkloněná hlava dopředu (spánek), která způsobí anomální pózu všech obličejových bodů. Většinou navíc dojde k výpadku detektoru tváří, kterému předchází anomální sekvence, což způsobí, že jsou všechny takové snímky označeny za anomální. Úspěšnost detekce této třídy je 81,02 %. Spánek se zakloněnou hlavou dozadu, případně do boku je detekován s úspěšností 66,94 %. Nižší procento je způsobeno tím, že pokud je hlava zakloněná pouze mírně dozadu (je opřená o opěrku hlavy) není často póza dostatečně anomální na to, aby ji VAE špatně rekonstruoval. Pokud by ovšem v reálné situaci došlo k usnutí nebo ztrátě vědomí řidiče, je pravděpodobné, že hlava spadne do strany či dopředu, což aplikace detekuje úspěšně. Další anomálií je kašel, který je detekován s úspěšností 72,06 %. Pokud se jedná o pouhý kašel, opět tím není způsobený anomální pohyb ani póza hlavy. V případě silného kašle či dušení se řidiče je ale metoda úspěšná. Nejhůře aplikace detekuje telefonování a to pouze v 22,21 %. Telefonování totiž nelze z orientačních bodů obličeje rozeznat. Aplikace je tedy úspěšná spíše výjimečně a to v případech, kdy je způsoben anomální pohyb hlavy, nebo špatná predikce orientačních bodů v důsledku telefonu, který zakrývá část tváře.

  Důležitým aspektem při hodnocení aplikace jsou také špatné detekce. Jejich počet je u testovacích dat následující:

  – Falešně pozitivní detekce: 1010

– Falešně negativní detekce: 1416

Do falešně negativních detekcí nejvíce přispívá špatná detekce telefonování, která celkem způsobuje 550 falešně negativních snímků, což tvoří 38,84 %. Větším problémem jsou ale falešně pozitivní detekce. Počet po sobě jdoucích snímků falešně označených za anomální je v rozsahu od jednoho (22 případů) až po 14 (2 případy), medián počtu je roven třem. To ukazuje, že falešně detekované anomálie jsou ve většině případů relativně krátké. To může být v některých případech způsobeno tím, že metoda detekuje začátek a konec anomálie jinak, než je to uvedeno v souboru s anotacemi. Počet sekvencí po sobě jdoucích snímků falešně označených za pozitivní, kterým předchází, nebo je následuje skutečná anomálie, je 42 a celkem je tvoří 114 snímků. Určit přesný začátek a konec anomálie je subjektivní a aplikace může být v některých případech přesnější než vytvořené anotace. Mezní hodnota je nastavená tak, aby falešně pozitivních, tedy anomálních detekcí bylo 10 % ze všech normálních snímků. Vzhledem k tomu, že při falešné detekci anomálie by byl při reálném provozu aplikace řidič zbytečně vyrušován, je to stále vysoké procento. Pro nižší hodnoty ale úspěšnost modelu poměrně strmě klesá. Při míře falešně pozitivních detekcí 7,5 % je úspěšně detekováno 62,79 % anomálních snímků, při 5 % už je to pouhých 38,47 %. Špatná detekce některých tříd anomálií a falešně pozitivní detekce tak představují hlavní problémy navržené aplikace.

Kapitola 5

Závěr

Detekce anomálních stavů řidiče by mohla výrazně přispět k bezpečnosti na cestách. Systém by na základě anomálního skóre a délky probíhané anomálie vyhodnotil vážnost situace a automobil by mohl adekvátně zareagovat zvukovou výstrahou, nebo přebráním řízení. Pro lepší vyhodnocení situace by bylo přínosné propojení detektoru s ostatními systémy vozidla. Dobře spolupracující systémy by mohly analyzovat styl jízdy a stav řidiče současně a tím zvyšovat možnost na úspěšné zamezení nebezpečí a zároveň minimalizovat falešné detekce. Systém by měl být schopný bezpečně detekovat především život ohrožující anomálie, jako jsou spánek, nebo ztráta vědomí.

Aplikace navržená v rámci této diplomové práce kombinuje metody pro detekci a popis tváře s použitím variačního autoenkodéru pro detekci anomálií. Je schopná ve většině případů úspěšně detekovat pravděpodobně nejnebezpěčnější anomální stavy řidiče vozidla, kterými jsou spánek a ztráta vědomí. Hlavní problém aplikace jsou pak falešné detekce, které by měly být co nejvíce minimalizované. Výsledná aplikace analyzuje video v reálném čase pouze na grafické kartě. Vzhledem k výpočetním limitům zabudovaných systémů by tedy nemohla být ve vozidle použitá. Dosažené výsledky však ukazují, že asistenční systém tohoto typu by mohl být realizovaný.

Při použití orientačních bodů obličeje je rozsah anomálií, které mohou být detekované, zúžený pouze na takové, které způsobí anomální pohyb, nebo pózu tváře. Jejich predikce je navíc časově nejnáročnější částí aplikace. Pro dosažení lepších výsledků by proto bylo vhodné zvážit jinou reprezentaci obličeje, např. použití HOG deskriptoru, nebo výstupu CNN. Variační autoenkodér se naopak ukázal jako vhodná architektura pro detekci anomálií. K dosažení lepších výsledků by ale měl být natrénovaný na co největším počtu sekvencí, které by měly zachycovat co nejvíce různých řidičů v normálních situacích.

V oblasti výzkumu zaměřené na sledování stavu řidiče vozidla bylo zatím publikováno jen málo prací, které se navíc většinou zabývají detekcí pouze určitých stavů. Pozornost a zdravotní stav řidiče jsou přitom bezpochyby jedněmi z nejdůležitějších aspektů bezpečnosti na cestách. Vzhledem ke stále se zlepšujícím systémům automobilů a rozvoji asistenčních funkcí pro zvýšení bezpečnosti se dá očekávat vývoj aplikací na toto téma jak ze strany výzkumných institucí, tak výrobců automobilů.

Literatura

1. *Road Safety* [online]. 2020-12-21 [cit. 2020-12-30]. Dostupné z: https://www.who.int/data/gho/data/themes/road-safety.
2. PAPAGEORGIOU, C. P.; OREN, M.; POGGIO, T. A general framework for object detection. In: *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*. 1998, s. 555–562. Dostupné z DOI: 10.1109/ICCV.1998.710772.
3. VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001.* 2001, sv. 1, s. I–I. Dostupné z DOI: 10.1109/CVPR.2001.990517.
4. VIOLA, P.; JONES, M.; SNOW, D. Detecting pedestrians using patterns of motion and appearance. In: *Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003)*. 2003, sv. 2, s. 734–741.
5. MOHAN, A.; PAPAGEORGIOU, C.; POGGIO, T. Example-based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2001, roč. 23, č. 4, s. 349–361. Dostupné z DOI: 10.1109/34.917571.
6. HOANG, V.D.; VAVILIN, A.; JO, K.H. Pedestrian detection approach based on modified haar-like features and adaboost. *International Conference on Control-Automation and Systems (ICCAS2012)*. 2012, s. 614–618.
7. SULEIMAN, A.; CHEN, Y.; EMER, J.; SZE, V. Towards closing the energy gap between HOG and CNN features for embedded vision. In: *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2017, s. 1–4. Dostupné z DOI: 10.1109/ISCAS.2017.8050341.
8. KOBAYASHI, T.; HIDAKA, A.; KURITA, T. Selection of Histograms of Oriented Gradients Features for Pedestrian Detection. *Neural Information Processing (CONIP 2007)*. 2008, 598–607.
9. FELZENSZWALB, P.F; MCALLESTER, D.A.; RAMANAN, D. A Discriminatively Trained, Multiscale, Deformable Part Model. In: 2008-06, sv. 8: dostupné z DOI: 10.1109/CVPR.2008.4587597.

10. FELZENSZWALB, P. F.; GIRSHICK, R. B.; MCALLESTER, D.; RAMANAN, D. Object Detection with Discriminatively Trained Part-Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2010, roč. 32, č. 9, s. 1627–1645. Dostupné z DOI: 10.1109/TPAMI.2009.167.

11. GANDHI, R. *Support Vector Machine — Introduction to Machine Learning Algorithms* [online]. 2018-06-07 [cit. 2021-02-03]. Dostupné z: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47.

12. JIAO, L.; ZHANG, F.; LIU, F.; YANG, S.; LI, L.; FENG, Z.; QU, R. A Survey of Deep Learning-based Object Detection. *CoRR*. 2019, roč. abs/1907.09408. Dostupné z arXiv: 1907.09408.

13. HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep Residual Learning for Image Recognition. *CoRR*. 2015, roč. abs/1512.03385. Dostupné také z: http://arxiv.org/abs/1512.03385.

14. KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In: PEREIRA, F.; BURGES, C. J. C.; BOTTOU, L.; WEINBERGER, K. Q. (ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2012, sv. 25, s. 1097–1105. Dostupné také z: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

15. SIMONYAN, K.; ZISSERMAN, A. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. Dostupné z arXiv: 1409.1556 [cs.CV].

16. HOWARD, A. G.; ZHU, M.; CHEN, B.; KALENICHENKO, D.; WANG, W.; WEYAND, T.; ANDREETTO, M.; ADAM, H. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. 2017. Dostupné z arXiv: 1704.04861 [cs.CV].

17. LIN, T.-Y.; MAIRE, M.; BELONGIE, S. J.; BOURDEV, L. D.; GIRSHICK, R. B.; HAYS, J.; PERONA, P.; RAMANAN, D.; DOLLÁR, P.; ZITNICK, C. L. Microsoft COCO: Common Objects in Context. *CoRR*. 2014, roč. abs/1405.0312. Dostupné také z: http://arxiv.org/abs/1405.0312.

18. GIRSHICK, R.; DONAHUE, J.; DARRELL, T.; MALIK, J. *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2014. Dostupné z arXiv: 1311.2524 [cs.CV].

19. EVERINGHAM, M.; VAN GOOL, L.; WILLIAMS, C. K. I.; WINN, J.; ZISSERMAN, A. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*. 2009, roč. 88, s. 303–308.

20. GIRSHICK, R. Fast R-CNN. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, s. 1440–1448. Dostupné z DOI: 10.1109/ICCV.2015.169.

21. REN, S.; HE, K.; GIRSHICK, R.; SUN, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2017, roč. 39, č. 6, s. 1137–1149. Dostupné z DOI: 10.1109/TPAMI.2016.2577031.

22. LIU, L.; OUYANG, W.; WANG, X.; FIEGUTH, P. W.; CHEN, J.; LIU, X.; PIETIKÄINEN, M. Deep Learning for Generic Object Detection: A Survey. *CoRR*. 2018, roč. abs/1809.02165. Dostupné také z: http://arxiv.org/abs/1809.02165.

23. LIU, W.; ANGUELOV, D.; ERHAN, D.; SZEGEDY, C.; REED, S.; A.C BERG, C.-Y. Fu ang. SSD: Single Shot MultiBox Detector. *Lecture Notes in Computer Science*. 2016, 21–37. ISBN 9783319464480. ISSN 1611-3349. Dostupné z DOI: 10.1007/978-3-319-46448-0_2.

24. REDMON, J.; DIVVALA, S.; GIRSHICK, R.; FARHADI, A. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. Dostupné z arXiv: 1506.02640 [cs.CV].

25. REDMON, J.; FARHADI, A. YOLO9000: Better, Faster, Stronger. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, s. 6517–6525. Dostupné z DOI: 10.1109/CVPR.2017.690.

26. BOCHKOVSKIY, A.; WANG, Ch.-Y.; LIAO, H.-Y. M. *YOLOv4: Optimal Speed and Accuracy of Object Detection*. 2020. Dostupné z arXiv: 2004.10934 [cs.CV].

27. WANG, C.-Y.; LIAO, H.-Y. M.; YEH, I.-H.; WU, Y.-H.; CHEN, P.-Y.; HSIEH, J.-W. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. *CoRR*. 2019. Dostupné také z: http://arxiv.org/abs/1911.11929.

28. CAO, Z.; SIMON, T.; WEI, S.-E.; SHEIKH, Y. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *CoRR*. 2016, roč. abs/1611.08050. Dostupné také z: http://arxiv.org/abs/1611.08050.

29. CAO, Z.; HIDALGO, G.; SIMON, T.; WEI, S.-E.; SHEIKH, Y. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *CoRR*. 2018, roč. abs/1812.08008. Dostupné také z: http://arxiv.org/abs/1812.08008.

30. TANUGRAHA, P. *Understanding OpenPose (with code reference)— Part 1* [online]. 2019-10-13 [cit. 2021-02-09]. Dostupné z: https://medium.com/analytics-vidhya/understanding-openpose-with-code-reference-part-1-b515ba0bbc73.

31. 300 Faces In-The-Wild Challenge: database and results. *Image and Vision Computing*. 2016, roč. 47, s. 3–18. ISSN 0262-8856. Dostupné z DOI: https://doi.org/10.1016/j.imavis.2016.01.002. 300-W, the First Automatic Facial Landmark Detection in-the-Wild Challenge.

32. KAZEMI, V.; SULLIVAN, J. One millisecond face alignment with an ensemble of regression trees. *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, s. 1867–1874.

33. *Real-Time Face Pose Estimation* [online]. 2014-08-28 [cit. 2021-02-10]. Dostupné z: http://blog.dlib.net/2014/08/real-time-face-pose-estimation.html.

34. BULAT, A.; TZIMIROPOULOS, G. How far are we from solving the 2D & 3D Face Alignment problem? (and a dataset of 230,000 3D facial landmarks). In: *International Conference on Computer Vision*. 2017.

35. NEWELL, A.; YANG, K.; DENG, J. Stacked Hourglass Networks for Human Pose Estimation. *CoRR*. 2016, roč. abs/1603.06937. Dostupné také z: http://arxiv.org/abs/1603.06937.

36. BULAT, A.; TZIMIROPOULOS, G. Binarized Convolutional Landmark Localizers for Human Pose Estimation and Face Alignment with Limited Resources. *CoRR*. 2017, roč. abs/1703.00862. Dostupné také z: http://arxiv.org/abs/1703.00862.

37. CHUNG, J.; GULCEHRE, C.; CHO, K.; BENGIO, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *CoRR*. 2014, roč. abs/1412.3555. Dostupné také z: http://arxiv.org/abs/1412.3555.

38. ROCCA, J. *Understanding Variational Autoencoders (VAEs)* [online]. 2019-09-24 [cit. 2021-02-26]. Dostupné z: https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73.

39. ODAIBO, S. G. Tutorial: Deriving the Standard Variational Autoencoder (VAE) Loss Function. *CoRR*. 2019, roč. abs/1907.08956. Dostupné také z: http://arxiv.org/abs/1907.08956.

40. SHAFKAT, I. *Intuitively Understanding Variational Autoencoders* [online]. 2018-02-04 [cit. 2021-02-26]. Dostupné z: https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf.

41. CHALAPATHY, R.; CHAWLA, S. Deep Learning for Anomaly Detection: A Survey. *CoRR*. 2019, roč. abs/1901.03407. Dostupné také z: http://arxiv.org/abs/1901.03407.

42. NAYAK, R.; PATI, U. Ch.; DAS, S. K. A comprehensive review on deep learning-based methods for video anomaly detection. *Image and Vision Computing*. 2021, roč. 106, s. 104078. ISSN 0262-8856. Dostupné z DOI: https://doi.org/10.1016/j.imavis.2020.104078.

43. REDMON, J.; FARHADI, A. YOLOv3: An Incremental Improvement. *CoRR*. 2018. Dostupné z arXiv: 1804.02767.

44. MORAIS, R.; LE, V.; TRAN, T.; SAHA, B.; MANSOUR, M. R.; VENKATESH, S. Learning Regularity in Skeleton Trajectories for Anomaly Detection in Videos. *CoRR*. 2019. Dostupné také z: http://arxiv.org/abs/1903.03295.

45. CHONG, Y. S.; TAY, Y. H. Abnormal Event Detection in Videos using Spatiotemporal Autoencoder. *CoRR*. 2017, roč. abs/1701.01546. Dostupné také z: http://arxiv.org/abs/1701.01546.

46. LIU, W.; LUO, W.; LIAN, D.; GAO, S. Future Frame Prediction for Anomaly Detection – A New Baseline. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018-06.

47. RONNEBERGER, O.; FISCHER, P.; BROX, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Springer International Publishing, 2015, s. 234–241.

48. CHEN, D.; WANG, P.; YUE, L.; ZHANG, Y.; JIA, T. Anomaly detection in surveillance video based on bidirectional prediction. *Image and Vision Computing*. 2020, roč. 98, s. 103915. ISSN 0262-8856. Dostupné z DOI: https://doi.org/10.1016/j.imavis.2020.103915.

49. KÖPÜKLÜ, O.; ZHENG, J.; XU, H.; RIGOLL, G. *Driver Anomaly Detection: A Dataset and Contrastive Learning Approach*. 2020. Dostupné z arXiv: 2009.14660 [cs.CV].

50. YOU, F.; GONG, Y.; TU, H.; LIANG, J.; WANG, H. A Fatigue Driving Detection Algorithm Based on Facial Motion Information Entropy. *Journal of Advanced Transportation*. 2020. Dostupné z DOI: https://doi.org/10.1155/2020/8851485.

51. QING, W.; BING-XI, S.; BIN, X.; JUNJIE, Z. A PERCLOS-Based Driver Fatigue Recognition Application for Smart Vehicle Space. *2010 Third International Symposium on Information Processing*. 2010, s. 437–441.

52. RONG-BEN, W.; LIE, G.; BINGLIANG, T.; LI-SHENG, J. Monitoring mouth movement for driver fatigue or distraction with one camera. *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No.04TH8749)*. 2004, s. 314–319.

53. GUPTA, R.; AMAN, K.; SHIVA, N.; SINGH, Y. An improved fatigue detection system based on behavioral characteristics of driver. In: *2017 2nd IEEE International Conference on Intelligent Transportation Engineering (ICITE)*. 2017, s. 227–230. Dostupné z DOI: 10.1109/ICITE.2017.8056914.

54. KHAN, M. Q.; LEE, S. A Comprehensive Survey of Driving Monitoring and Assistance Systems. *Sensors*. 2019, roč. 19, č. 11. ISSN 1424-8220. Dostupné z DOI: 10.3390/s19112574.

55. KING, D. E. Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research*. 2009, roč. 10, s. 1755–1758.

56. PONNUSAMY, A. *cvlib - high level Computer Vision library for Python* [https://github.com/arunponnusamy/cvlib]. 2018.

57. BRADSKI, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*. 2000.

58. HUANG, Gary B.; RAMESH, Manu; BERG, Tamara; LEARNED-MILLER, Erik. *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*. 2007. Tech. zpr., 07-49. University of Massachusetts, Amherst.

59. GUPTA, V. *Face Detection – OpenCV, Dlib and Deep Learning (C++ / Python)* [online]. 2018-10-22 [cit. 2021-02-26]. Dostupné z: `https://learnopencv.com/face-detection-opencv-dlib-and-deep-learning-c-python/`.

60. NGUYEN, T. *Deep learning based Face detection using the YOLOv3 algorithm* [online]. 2018 [cit. 2021-02-26]. Dostupné z: `https://github.com/sthanhng/yoloface/tree/2b954f318d9bd9136836bed0a71109ab56681790`.

61. YANG, S.; LUO, P.; LOY, Ch. Ch.; TANG, X. WIDER FACE: A Face Detection Benchmark. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

62. CHOLLET, F. et al. *Keras*. 2015. Dostupné také z: `https://keras.io`.

63. ABADI, Martín; AGARWAL, Ashish; BARHAM, Paul; BREVDO, Eugene; CHEN, Zhifeng; CITRO, Craig; CORRADO, Greg S.; DAVIS, Andy; DEAN, Jeffrey; DEVIN, Matthieu; GHEMAWAT, Sanjay; GOODFELLOW, Ian; HARP, Andrew; IRVING, Geoffrey; ISARD, Michael; JIA, Yangqing; JOZEFOWICZ, Rafal; KAISER, Lukasz; KUDLUR, Manjunath; LEVENBERG, Josh; MANÉ, Dandelion; MONGA, Rajat; MOORE, Sherry; MURRAY, Derek; OLAH, Chris; SCHUSTER, Mike; SHLENS, Jonathon; STEINER, Benoit; SUTSKEVER, Ilya; TALWAR, Kunal; TUCKER, Paul; VANHOUCKE, Vincent; VASUDEVAN, Vijay; VIÉGAS, Fernanda; VINYALS, Oriol; WARDEN, Pete; WATTENBERG, Martin; WICKE, Martin; YU, Yuan; ZHENG, Xiaoqiang. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Dostupné také z: `https://www.tensorflow.org/`.